Universidade de Aveiro Departamento de Engenharia Mecânica
2018

Diogo Filipe Julião da Rocha

Proposal of an automatic traceability system, in Industry 4.0

Proposta de um sistema automático de rastreabilidade, na indústria 4.0

Universidade de Aveiro Departamento de Engenharia Mecânica
2018

Diogo Filipe Julião da Rocha

Proposal of an automatic traceability system, in Industry 4.0

Proposta de um sistema automático de rastreabilidade, na indústria 4.0

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestrado em Engenharia Mecânica, realizada sob orientação científica de José Paulo Oliveira Santos, Professor Auxiliar da Universidade de Aveiro e de Abílio Manuel Ribeiro Borges, Assistente Convidado do Departamento de Engenharia Mecânica da Universidade de Aveiro.

Com o apoio dos projetos:
UID/EMS/00481/2013-FCT e
CENTRO-01-0145-FEDER-022083

O júri / The jury

Presidente / President

Prof. Doutor Jorge Augusto Fernandes Ferreira
Professor Auxiliar da Universidade de Aveiro

Vogais / Committee

Prof. Doutor José Paulo Oliveira Santos
Professor Auxiliar da Universidade de Aveiro (orientador)

Profª. Doutora Ana Maria Pinto de Moura
Professor Auxiliar da Universidade de Aveiro

Agradecimentos / Acknowledgements

Em primeiro lugar quero agradecer aos meus pais e irmão pelo apoio que sempre me deram. Aos meus amigos e colegas que me apoiaram e ajudaram durante o meu percurso académico e o tornaram memorável. Ao Prof. Doutor José Paulo Santos por ter orientado a minha dissertação, e por sempre ter mostrado disponibilidade qualquer que fosse o problema. Ao Prof. Abílio Borges por me ter orientado e pelos sua indespensável ajuda em fazer a ponte com a Renault CACIA. E ao Eng. Contantino Pinto da Renault CACIA por se ter mostrado disponível para explicar o funcionamento da linha de carters.

Keywords Traceability; Industry 4.0; AIDC; RFID; RWM; Tag; ESP32; RS485; Database; web services; Graphical interface

Abstract Traceability systems are very valuable to manufacturing companies, allowing to keep detailed records of its products history. The advent of Industry 4.0 further pushes this concept, integrating the Internet of Things into traceability by creating smart objects capable of communicating through the internet. RFID is the technology with the most potential for traceability due to its several advantages over technologies such as barcodes. The housings production line of Renault CACIA was used as the example of a shop floor on which to apply the proposed traceability system. This dissertation's system uses a cost effective microchip, the ESP32, as the middleware that processes the messages exchanged with RFID Read/Write Modules, which are placed in several of the production line's stations. The Wi-Fi capabilities of ESP32 are also used to communicate the data with a server, which processes the data and performs queries to a MySQL database. This database was normalized, and applies specifically to the housings production line, allowing to keep detailed records of every housing produced on the shop floor. A web graphical interface was developed, with its pages showing both real time data as well as historical production data. These web services can be accessed by any IoT device capable of accessing to the server. The developed solution showcases the viability of chips such as ESP32 to be integrated in RFID systems, allowing RFID readers and tags to be able to communicate through the internet.

Palavras-chave

Rastreabilidade; Indústria 4.0; AIDC; RFID; RWM; Tag; ESP32; RS485; Base de dados; Serviços web; Interface gráfica

Resumo

Os sistemas de rastreabilidade são valiosos para as empresas fabricantes, permitindo manter um registo do histórico dos seus produtos. O surgimento da Indústria 4.0 avança ainda mais este conceito, integrando a *Internet of Things* na rastreabilidade ao criar objetos inteligentes capazes de comunicar pela Internet. A tecnologia RFID é a tecnologia com mais potencial na rastreabilidade devido às suas diversas vantagens quando comparada a tecnologias como os códigos de barra. A linha de produção de carters da Renault CACIA foi utilizada como um exemplo de um chão de fábrica no qual se aplica o sistema de rastreabilidade proposto. O sistema desta dissertação utiliza um microcontrolador de baixo custo, o ESP32, como o *middleware* responsável por processar as mensagens trocadas com os leitores RFID, sendo estes colocados em vários postos da linha de produção. As capacidades Wi-Fi do ESP32 são utilizadas para comunicar com um server que processa os dados e faz *queries* a uma base de dados MySQL. Esta base de dados foi normalizada e aplica-se especificamente à linha de carters, permitindo manter registos detalhados sobre cada carter produzido na fábrica. Foi desenvolvida uma interface gráfica web, cujas páginas mostram informação em tempo real sobre a linha, assim como informação sobre o histórico da produção. Estes serviços web podem ser acedidos por qualquer equipamento da IoT capaz de aceder ao server. A solução desenvolvida demonstra a viabilidade da integração de chips como o ESP32 em sistemas de RFID, permitindo que leitores e etiquetas RFID comuniquem pela Internet.

Contents

List of Tables

List of Figures

Acronyms

ADC - Analogic to Digital Converter

AGV - Automated Guided Vehicle

AIDC - Automatic Identification and Data Capture

AJAX - Asynchronous Javascript and XML

AT - Atelier

BCNF - Boyce-Codd Normal Form

BOM - Bill Of Materials

CAD - Computer-Aided Design

CED - Clutch Housing

CM - Mechanism Housing

CSS - Cascading Style Sheets

DAC - Digital to Analogic Converter

EAN - European Article Number

EIS - Enterprise Information Systems

EPC - Electronic Product Code

EPCIS - Eletronic Product Code Information Systems

ER - Entity Relationship

ERD - Entity Relationship Diagram

ERP - Enterprise Resource Planning

FDD - Functional Dependency Diagrams

GALIA - Groupement pour l'Amélioration des Liaisons dans l'Industrie Automobile

GLN - Global Location Number

GPIO - General Purpose Input/Output

GSM - Global System for Mobile Communications

GTIN - Global Trade Item Number

HF - High Frequency

HMI - Human Machine Interface

HTML - Hypertext Markup Language

I/O - Input/Output

I2C - Inter-Integrated Circuit

I2S - Inter-IC Sound

IC - Integrated Circuit

IoMT - Internet of Manufacturing Things

IoT - Internet of Things

JIT - Just-in-Time

JS - JavaScript

JSON - JavaScript Object Notation

LF - Low Frequency

NC - Nonconform

PC - Personal Computer

PCB - Printed Circuit Board

PLC - Power Line Communication

PLS - Pallet Loading Scheme

POE - Pièces Ouvrées à l'Extérieur (Externally Worked Parts)

POI - Pièces Ouvrées à l'Intérieur (Internally Worked Parts)

POS - Points-of-sale

PROFIBUS - Process Field Bus

PSFP - Pilotage et Suivi des Fluxs pièces (Control and Monitoring of Part's Flow

QA - Quality Assurance

QC - Quick Change

QRCode - Quick Response Codes

RFID - Radio Frequency Identification

RT-SMDS - Real-Time Shop-Floor Material Distribution System

RWM - Read/Write Module

SAW - Surface Acoustic Wave

SGTIN - Serialized Global Trade Item Number

SOA - Service Oriented Architecture

SPI - Serial Peripheral Interface

SSCC - Serial Shipping Container Code

SSID - Service Set Identifier

TCP/IP - Transmission Control Protocol/ Internet Protocol

UART - Universal Asynchronous Receiver-Transmitter

UEXT - Universal EXTension

UHF - Ultra High Frequency

UID - Unique Identifier

UML - Unified Modeling Language

UPC - Universal Product Code

UTC - Coordinated Universal Time

VDC - Voltage Direct Current

XML - Extensible Markup Language

Chapter 1

Introduction

1.1 Context

Traceability is a core need of the manufacturing industry, allowing to control, supervise and access the history of a company's resources. The advent of Industry 4.0 has further advanced the concept of traceability, proposing that each object of a supply chain should be tracked at all times. Furthermore, all this data should be readily available through enterprise information systems (EIS) to all participants of a supply chain. One key enabling technology of traceability is Radio Frequency Identication (RFID), allowing to quickly identify objects, and consequently make traceability an automatic process [1]. However, several companies still adopt partially or fully manual traceability systems, which are prone to high occurrence of mistakes, low result confidence and slow update rate of their data. The solution is utilizing automatic traceability systems, complemented with web services, with the final purpose of helping companies improve their organization, improve their capacity to solve problems and above of all, boost their competitiveness.

The automotive sector in particular is highly competitive and complex. Car companies usually have multiple facilities handling the production of different components, coupled with the fact that the assembly of different car systems can occur in different facilities altogether. This naturally imposes the necessity of traceability systems, due to logistical, financial and quality assurance needs. Moreover, an adequate traceability solution must be in place to allow tracing back flawed components, getting to the cause of a defect and acting accordingly so that the mistake is not propagated to multiple batches of components.

This dissertation presents a traceability solution for the Renault CACIA gear housings production line. Information was provided by Renault CACIA and Atena - Automação Industrial Lda, the latter being responsible for implementing several of the already existing traceability systems in the line. Furthermore, a master thesis by F. Pinho [2] served as a starting point to better understand the production line, especially in the initial stages of the dissertation, when communication with Renault was starting to be established. The present dissertation sets out to create a traceability system that utilizes cost-effective IoT technologies while meeting the traceability needs of Renault CACIA.

1.2 Methodology and expected results

The objective of this dissertation is to propose and develop a traceability solution tailored specifically to the housings production line of Renault CACIA. In particular it must provide traceability of the products from the moment they enter the line to the their shipping to the assembly line. To achieve this, several stations should use the full potential of their already present RFID systems. Furthermore, each housing must be uniquely identified by a serial number, allowing each part to be tracked individually. To handle the communication with the RFID readers a middleware layer should be used. Ideally this layer should be of reduced cost, and able to communicate with a remote computer. It should also reduce the need for physical connections by using the Internet.

The remote computer is the top processing layer, and should communicate with a database present in the same central computer. This database will contain all the data related to the passage of objects through the workstations, and will also associate every housing with an arriving container and a departing container. It is also predicted the use of a graphical interface consisting of several web pages, allowing the production operators and managers to access real time information about the line.

In sum, the developed solution must take advantage of the Industry 4.0 philosophy, integrating different devices into a fully operational IoT application.

1.3 Dissertation organization

This dissertation is organized in six chapters. Chapter 1 gives a short introduction of the context and expected results of this dissertation. Chapter 2 explains the Renault CACIA gear housings production line. A brief introduction of Renault CACIA is given, followed up with a description of the several stations of the line. The RFID system currently implemented is explained, including its several components. An explanation of the possible nonconformities is given, concluding the introduction of the production line.

Chapter 3 presents the state of the art. The initial sections concern concepts surrounding the subject of this dissertation. It starts with an explanation of supply chains. Traceability and its importance are the subject of the next sections, followed by the definition of Industry 4.0. After this, traceability technologies are the focus of the next sections. Barcodes, both 1D and 2D, are explored, with comparisons between the different types of barcodes. Marking technologies are also looked at, such as laser marking and dot peen marking.

The traceability technology used in this dissertation, RFID, is given a more in-depth explanation in the next section of Chapter 3. A typical RFID system is explained, preceding information about the different tags in RFID systems, as well as the main communication methods between tags and readers. An account of the different frequencies associated with RFID communication is also given. A description of EPCglobal is given next, alongside some technologies GS1 uses alongside this standard. Next are shown several examples of traceability systems present in the literature, in both other dissertations and scientific articles. The chapter ends with brief summaries of some commercial traceability solutions.

Chapter 4 consists of the solution proposed in this dissertation. A general explanation of the processing algorithms is given next, preceding the proposal of a normalized

database. The database is conceptualized using Entity Relationship Diagrams, with each entity and relationship given a description.

Chapter 5 concerns the implementation of the proposed solution, with its intricacies further explored. The hardware is explained in detail. The assembly of the proposed system is shown in the following section, including the printed circuit built for this implementation. The data stored on the tags is explained before several diagrams are shown to help explain some of the algorithms developed. The last section illustrates several elements of the graphical interface.

Chapter 6 explains the conclusions reached in this dissertation, as well as some possible improvements for the developed traceability system.

Appendix A expands the previous explanation of the Graphical interface, with more in depth looks at all pages created. Appendix B details the data contained in the GALIA tags used at Renault. Appendix C shows the developed PCB's bottom and top layers in detail.

Chapter 2

Renault CACIA's housings production line

2.1 Introduction to Renault CACIA

Renault CACIA is one of 38 industrial facilities of the Groupe Renault, present in 17 countries [3]. The facility was founded in 1981, then called C.A.C.I.A. (Companhia Aveirense de Componentes para a Indústria Automóvel). It focused on gearbox production, having multiple clients at the time. Across the years its partnership with Renault grew, to the point that in 2001 it began producing solely to Groupe Renault, eventually changing its name to Renault CACIA. Figure 2.1 shows an aerial view of the facility.

Figure 2.1: Aerial view of Renault CACIA [4]

Presently the company produces several components for Renault and Nissan automobiles, including gearboxes, iron collectors and oil pumps. The company assembles some types of gearboxes, and produces the components for many others. This entails the manufacturing of shafts, gears and housings for gearboxes and differentials.

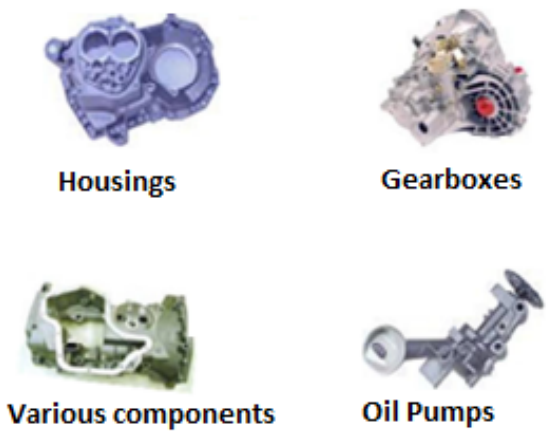


Figure 2.2: Products manufactured at Renault CACIA, adapted from [2]

The facility is divided into nine different departments, namely: Production, Engineering, Logistics, Quality Assurance, Financial, Technical, Information Technology, Production Systems and Human Resources. The Production Department is divided into five Ateliers, each specializing in a few components, or, in the case of Atelier 5 (AT5), the assembly of gearboxes. Figure 2.3 shows what each Atelier specializes in.

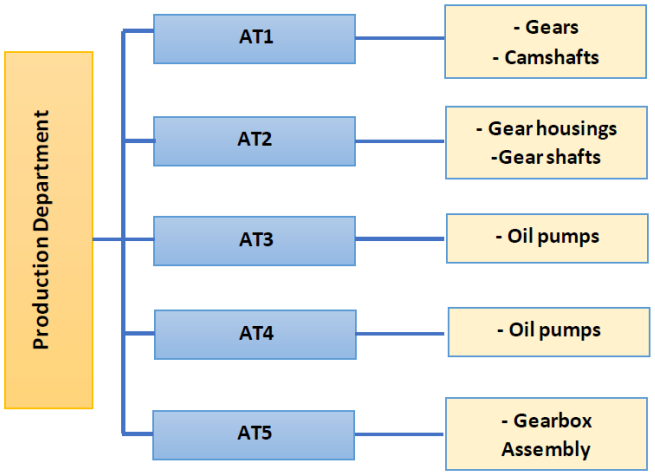


Figure 2.3: Organization of Renault's Production Department

2.2 The gear housings production lines

The subject of this thesis will be the gearbox housings production lines, located at Atelier 2. AT2 itself is divided into 7 production lines.

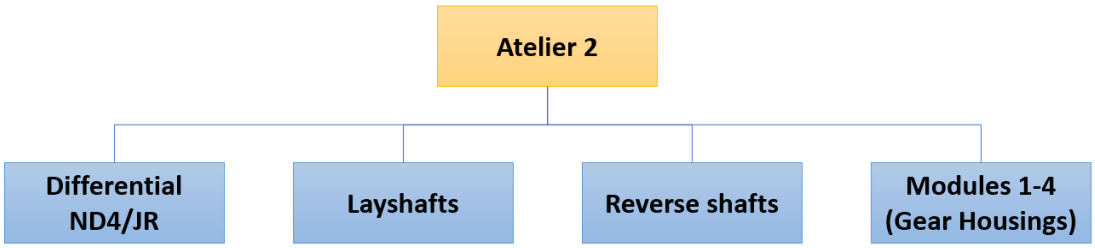


Figure 2.4: Products manufactured at Atelier 2

As seen in figure 2.4, two modules handle the manufacturing of gear shafts. One module produces layshafts, while the other produces only reverse shafts. Another module creates differential housings, with the differential being responsible for the distribution of power from the engine to the wheels. The other four modules exclusively manufacture gear housings. A housing provides the outer structure of a gearbox. Two types of gear housings are produced in AT2: clutch housings and mechanism housings, referred by Renault as CED and CM respectively. Both house the components of the gearbox, with the clutch housing covering the shaft that connects the engine and gearbox. Both gear housings, when assembled together, make a full gearbox. The two can be seen in figure 2.5.

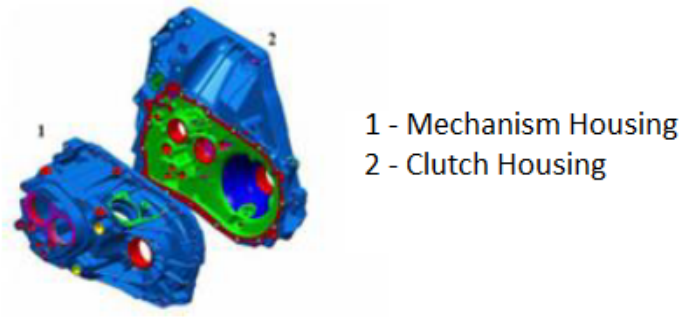


Figure 2.5: CAD of the two gearbox housings, adapted from [2]

2.3 Material of the gear housings

The gearbox housings are made of aluminium. This metal is widely used on the automotive industry due to its properties [5]: its low density and consequent light weight lead to better fuel performances from automobiles; its surface naturally consists of aluminum oxide layer, which protects the aluminum from further oxidation. This is particularly important for a gear housing, since it needs to be leak-proof. Finally, it has good physical properties, such as its ability to absorb shock energy [6]. In sum, aluminum gearboxes are common in the automotive sector, and Renault's housings are no exception. The Renault CACIA Production Department receives raw gearbox housings from five different suppliers: Fagor, Cléon, Alfisa, Vilanova and Dacia. The raw housings are produced through casting. However, this process doesn't create housings with the final product's dimensions. Renault CACIA will machine the cast aluminum housings and test their

quality, in order to later be assembled alongside all other gearbox components.

2.4 Gearbox housing's families

Each module of AT2 specializes in a certain family of gearbox housings. Several families are manufactured in Renault CACIA such as TL4, ND4 and JRQ. Among each family there are two types of housings, mechanism (CM) and clutch (CED), like the ones presented in figure 2.5. Some of the housings produced at Renault CACIA can be seen on figure 2.6.

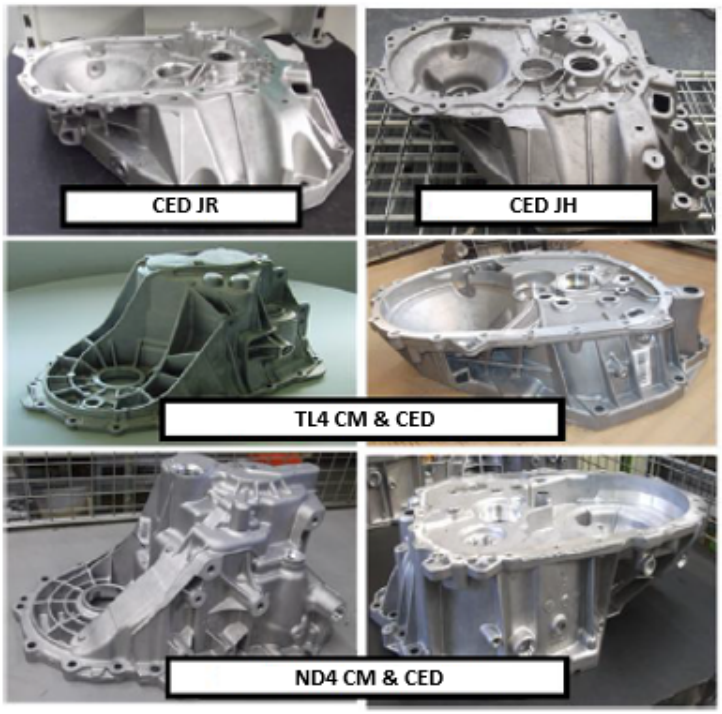


Figure 2.6: Types of housings produced at Renault CACIA [2]

CM housings present no variance within each family. CED housings, while very similar within each family, can have small differences between each other, which necessitates the creation of a new reference. Furthermore, within the same reference there can be differences. As an example, some JRQ CED housings can have a built-in tachometer, a device that measures the vehicle's velocity. This doesn't prompt the use of another reference entirely, despite the presence of a tachometer influencing the machining process and the final product the housing is destined for. Figure 2.7 shows the different references for the TL4, JRQ and ND4 housings families.

TL4	CM	CED						
Variety	--	K9K S2	M1G N	M1G R	F4R	K4M	H4J	K9K S3	H5FT

ND4	CM	CED	
Variety	--	F9Q	R9M

ND6	CM	CED	
Variety	--	YD	TR

Figure 2.7: Some housing references produced in Renault CACIA

2.5 Production line

A gear housing goes through several steps during its production at Renault CACIA. These are shown in the diagram of figure 2.8, and are explained in the following subsections.

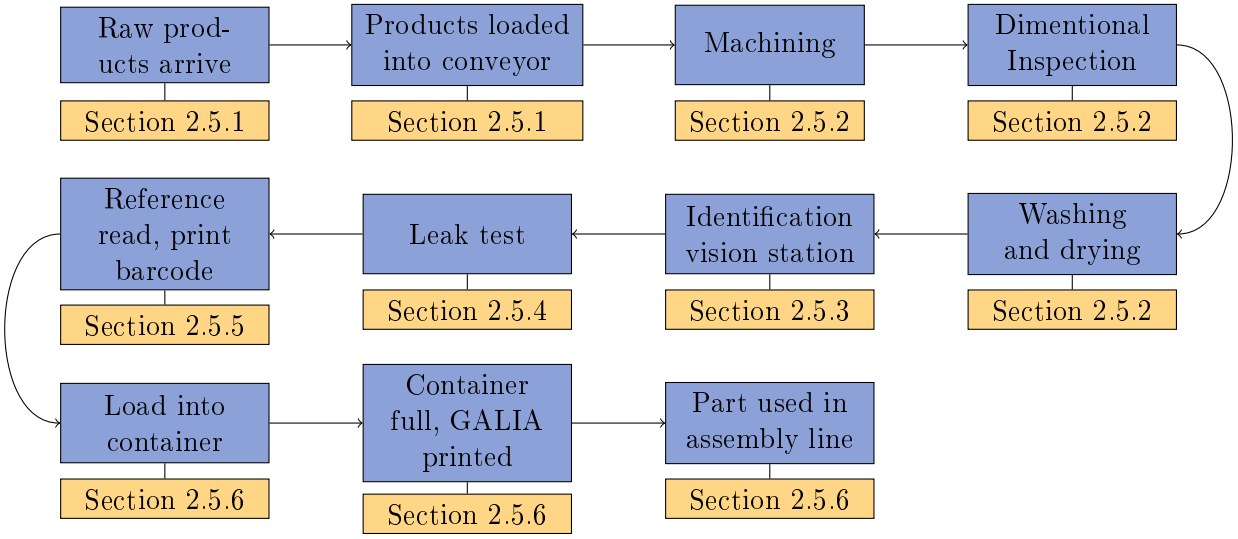


Figure 2.8: Diagram of Renault's gearbox housing's production line

Figure 2.9 shows the layout of module 3. The remaining modules have different layouts, though they contain the same workstations.

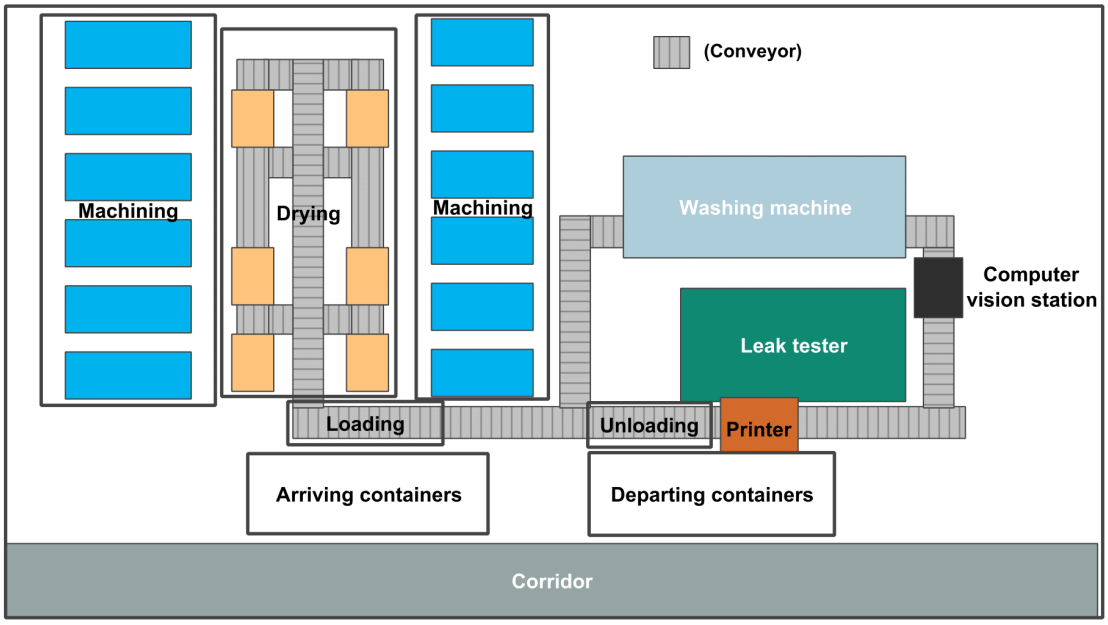


Figure 2.9: Production line (module 3) shop floor layout

2.5.1 Arrival of raw housings

At the beginning of a production line the raw cast housings are received, carried by containers. Each container is identified by a GALIA, a paper tag used by most french automotive companies. These tags hold the information necessary to trace lots that are distributed between different factories or departments. This data is used by the logistics department to plan the line's production. When an order is given for a lot to be machined, its raw products are loaded into the conveyor, as can be seen on figure 2.10. In this loading station a monitor displays which housing the operator should load at any given time.

Figure 2.10: Loading of housings to the conveyor [2]

A worker visually inspects each individual part, looking for defects, also called non-conformities. In case imperfections are detected, the part will go to Quality Assurance to further ascertain its quality, where it might be considered as junk if it fails further tests. The parts that pass the first visual inspection carry on to machining.

2.5.2 Machining centers

Parts that carry on to machining stop at a middle station, called the drying station. At Module 3 there is one drying station for every two machining centers. Firstly, operators write into the housing the machining center it will go into. The housing is then manually loaded into the machining center. Each object will go through vertical and horizontal machining. Firstly, two parts are vertically machined, followed by horizontal machining. The two machining centers used at Renault CACIA, from manufacturers DMC and GROB, are shown in figure 2.12. Between the vertical and horizontal machining the parts are quickly washed and dried.

Figure 2.11: Machining centers

Each shift or batch, a housing is subjected to a dimensional inspection, to make sure the machining procedure is producing housings with correct measures. However, normally, housings proceed from the machining to a washing machine, where the lubricant used in the machining is removed more thoroughly than in between the two machining operations.

2.5.3 Identification vision station

After being washed and dried, the housings are transported through the conveyor to a computer vision station. This station consists of a Genie Nano M2420 camera, 6 LED lights and an industrial computer. The computer is connected to the camera and runs a program developed in Sherlock, designed to identify the type of housing present in the image. This is achieved by searching for specific machined features of the housing that differentiate it from every other type. Each type of gearbox housing is identified by a reference, a ten digit number defined by Renault. For example, CM ND4 housings are identified by the number 8200667174.

Another purpose of this station is to store an image of the housing. A new folder is created everyday, itself containing several other folders, one for each reference. Furthermore, a text file keeps a log of all housings that go through the station, adding a new line for every housing. This line includes the date, the path to the stored image, the housing reference and the results of several tests done by the computer vision routine.

This is the primary system for keeping track of individual housings. The photo also contains the markings done by the operator before machining, which means this image can be used to trace back the machining center a housing went through.

2.5.4 Leak test

The reference of the housing obtained in the vision station is used in the leak test. A robot handles the housing during this operation, and for that it needs to know the kind of housing it is meant to handle.

If a housing has no leaks, its reference is stored in an RFID tag. The reference stored in the tag will later be used to print a barcode. If leaks were detected, the housing is removed from the production line through a different conveyor.

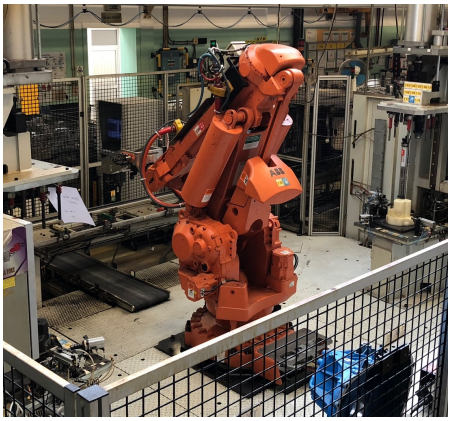


Figure 2.12: Leak tester station

2.5.5 Barcode printing

If a housing passed the test, its RFID tag is read and the object's reference is sent to a printer, which will print a barcode paper label containing the housing's designation (CM ND4, for example) and its corresponding reference number, as shown in figure 2.13.

Figure 2.13: Barcode attached to the housings

The print engines used in the housing production line are Zebra 110PAX4. They can print a variety of codes, both linear (EAN-8, EAN-13, GS-128) and 2D (DataMatrix and QR Code). It is also compatible with custom logos and can write in several fonts. Zebra printers can print RFID smart labels, however, for this operation the printer needs an RFID reader/encoder module installed.

Figure 2.14: Zebra 110PAX4 printer [7]

2.5.6 Shipping to assembly line

Housings are subjected to a final visual test by the operators, searching for specific nonconformities, which are explained in more detail in section 2.9. In case the housing passes this inspection, its barcode is scanned, indicating to Renault's information system that its production is completed. This information system is the PSFP, which allows, among other things, to create GALIA tags and insert them to Renault's database.

After being scanned, the housing is put on a container. When enough housings of the same reference are gathered to fill a container a GALIA tag is printed and attached to it.

Figure 2.15: Housings container

The container is then either shipped to another facility or used in AT5, where gearboxes are assembled. When it arrives at the assembly line the GALIA is read, mainly for logistical purposes. Each housing still carries the barcode that was applied in AT2. The code is read when it enters the assembly line, confirming the reference of the housing.

2.6 GALIA Tag

GALIA tags contain all the information related to a container, and are used by Renault and other french automotive companies. AT2 handles three types of GALIA's. Two of them identify products that arrive at the production line. These are associated with POE (Externally Worked Parts) and POI (Internally Worked Parts) products. POE are products manufactured by companies outside Groupe Renault, while POI are products manufactured by other Renault facilities.

The other possible GALIA tag identifies products leaving the gear housing production line. This type of tag is shown in figure 2.20.

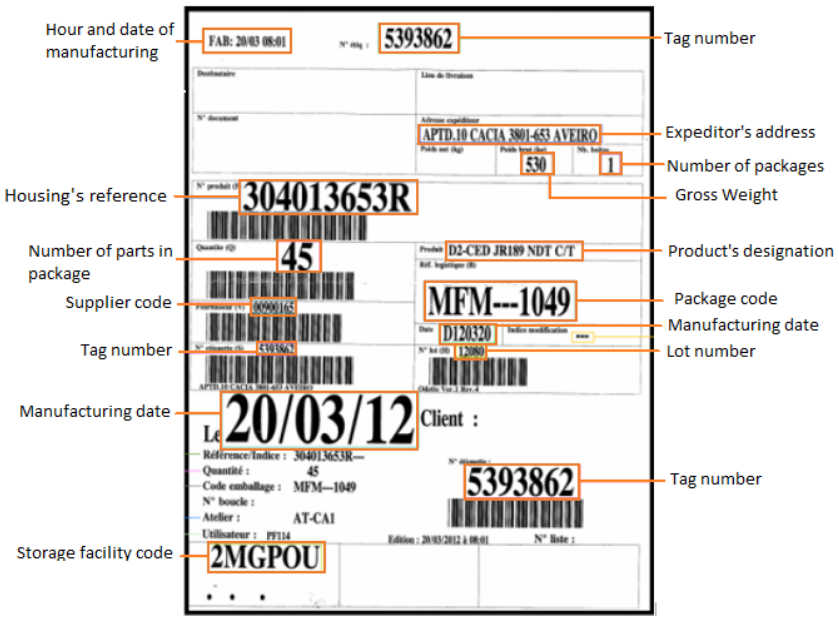


Figure 2.16: Finished products GALIA tag, adapted from [2]

As can be seen, it identifies several parameters. The most notorious are: date of manufacturing; reference of the gearbox housing; the storage facility it is destined for, the number of housings it contains; lot number; and the GALIA tag number itself.

The GALIA's information is meant to be read with a barcode scanner at the start of the assembly line. Some information, such as the date of manufacturing or the number of packages, has to be read by the operator handling the container and written manually on a traceability sheet.

2.7 Stratus database

Stratus is the database implemented by Renault, storing and allowing to search the history of components and products it manufactures. Figure 2.17 contains the search window of Stratus.

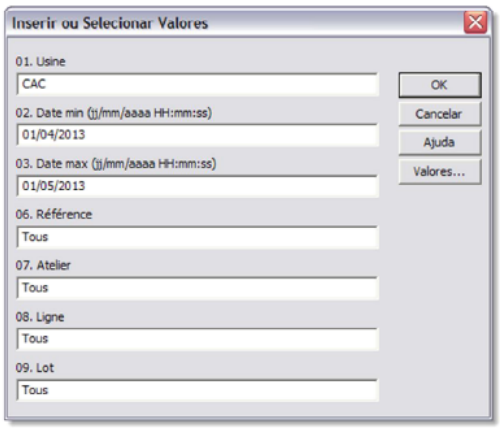


Figure 2.17: Stratus search window [8]

Stratus allows to search within certain parameters. It can narrow down the results according to an interval of dates, the reference of the housing, the Atelier, the production line and the lot. Depending on the parameters, the results will present different data. As an example, the following image shows the results after searching for the reference of CM ND4.

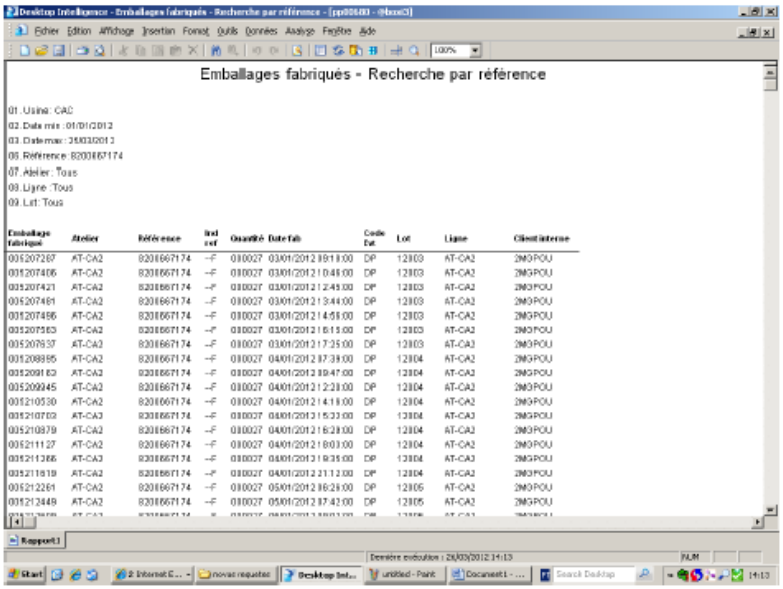


Figure 2.18: Results after searching for CM ND4 reference - 8200667174 [2]

Stratus shows the container's number (GALIA's number), the Atelier and module where it was manufactured, the part's reference, the quantity of gearbox housings, the manufacture date, the lot number and the storage facility it is currently in. 2MGPOU is where components for internal consumption are stored.

Noticeably this system does not keep track of individual components. It only keeps track of both arriving and departing containers, identifying them by their GALIA tag ID. The only system that keeps a detailed record of each housing is the computer vision station. Even then, this system is only valuable in the short term. In case there is a

problem with a gear housing after it is in use, it becomes nearly impossible to trace back the component to the machining center or casting mold it originated from.

2.8 RFID system

Renault CACIA already implements RFID technology in the housing production lines, employing hardware of the company Balogh. This system has three main components:

- Transceiver
- RFID tags
- Control interface.

At Renault the tags are attached to the pallets carrying the housing. RFID transceivers communicate with the tags, reading their memory, as well as writing data into the tags. The control interface, also called control board, manages the dialogue between the transceiver and tag. The control board accepts requests from a supervisor, a PC or a PLC, and send them to the RFID head. It will also process the responses given by the tag to those same requests.

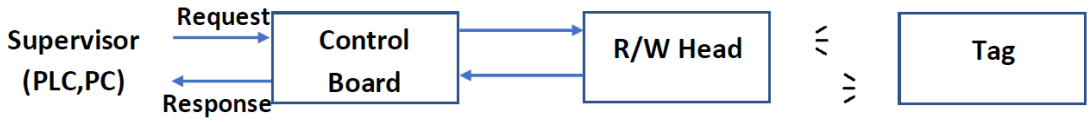


Figure 2.19: Renault CACIA RFID system schematic

2.8.1 ERC-85/QC Transceiver

This transceiver is specifically designed for industrial use, having a Rilsan casing and working at maximum temperatures of 70 °C. It has a maximum current consumption of 150 mA and needs a power supply of 24 VDC. A male QC (Quick Change) connector is available, connecting with the control interface.

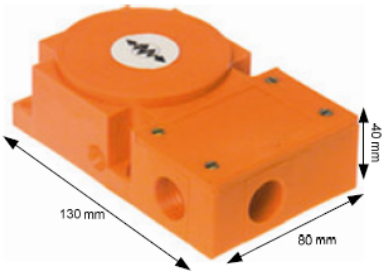


Figure 2.20: ERC-85/QC Transceiver and its dimensions [9]

2.8.2 OMX 931 Tags

These RFID tags communicate in the 13.56 MHz frequency, in accordance with the ISO 15693 standard. Their total memory is 8 kBytes and the maximum communication

range is 50 mm when paired the ERC-85 transceiver. They are passive tags, and like the transceiver are resistant to temperatures up to 70 °C. These tags are mounted on the metal pallets that carry the housings, and are especially designed to allow communication in such an environment. They can be mounted while recessed in metal as long as a clearance of at least 10 mm exists between the tag and the metal.

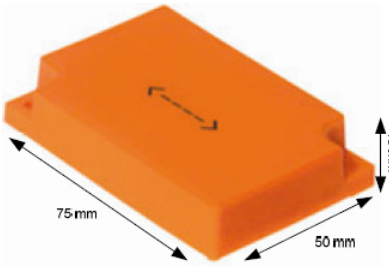


Figure 2.21: Balogh OMX 931 tag and its dimensions [9]

2.8.3 BIDP-170 DP Control Interface

The BIDP-170 control interfaces are used on Renault as the middleware between the supervisor equipments (PC and PCL's) and Read/Write heads. They interpret the tag's responses to the transceiver's requests. These requests are processed by this control interface as well. This board requires 24 V of direct current, consuming a total of 350 mA when connected to two transceivers. It operates at maximum temperature of 55 °C.

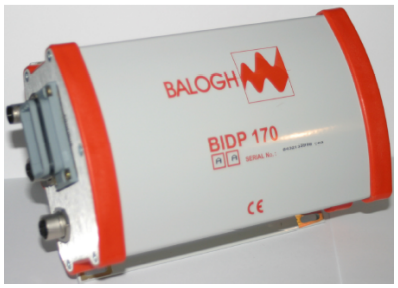


Figure 2.22: BIDP-170 DP Control Interface [10]

The control board allows simultaneous communication with a maximum of two transceivers. Figure 2.23 shows the two M12 5-pin sockets used for connecting the transceivers, named CHANNELS 1 and 2. The same figure shows the 24 VDC connector. And lastly, it illustrates the SUB-D 9 female connector, which uses the RS-485 protocol, utilized to communicate with the PROFIBUS network of Renault.

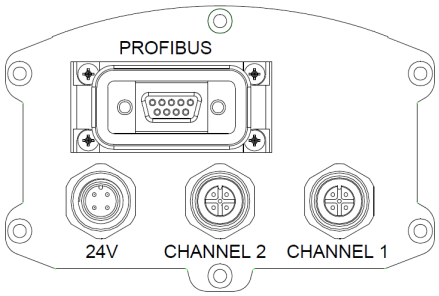


Figure 2.23: BIDP-170 DP Control Interface's connector flange [11]

PROFIBUS is a standard for fieldbus communications. It provides a common standard of communication for devices, no matter the device's manufacturer, and focuses in the area of automation. Specifically this control board uses PROFIBUS DP (Decentralized Peripherals). This is a high-speed solution for PROFIBUS [12] , with emphasis on operating on decentralized devices such as I/O's, motion controls, motor controls, motor starters and photo cells. These devices are the slaves in the communication, answering to commands from its masters (PC's or PLC's). The communication between masters and slaves occurs typically in cycles. However, some peripherals, namely the BIDP-170 DP Control Interface, allow non-cyclic communications, such as configuration or turn on/off messages.

Transceivers are located in several spots of the line: the entrance of the machining area, housing identification vision station, leak tester station and printer station.

2.8.4 Data stored in Tags

Presently the data stored in the Balogh RFID tags is the following:

Data	Abbreviation	Memory location	Value
Date	Date	0 to 7	Date of entry on system
Gearbox Type	TPBTE	8	1 if ND6 2 if ND0 3 etc.
Housing Type	TPCRT	9	1 CM 2 CED 9 empty
Variants	VARTE	10	1 YD 2 TR 3 etc.
Destination	DEST	11	1 to 6

Table 2.1: Data stored in Balogh tags

The first eight bytes of the memory are used to store the date the housing passed at the entrance of the machining area. A byte is used to identify the type of gearbox the housing is meant for. The next byte identifies the type of housing, between CM and CED. The housing is further identified by its variant. The tag also stores a byte indicating the drying station the housing is meant to go to, named Destination.

Currently the data stored in the tag presents little traceability value. Several of its bytes, especially the Destination byte, are used to automate the transportation of the housings across the conveyor. Furthermore, bytes 8 to 10, which identify the housing, are mostly used in the last stations: the reference is stored in the leak tester station and is subsequently read at the printer station in order for a paper tag to be applied on the housing.

2.8.5 Limitations of the current RFID system

There are a number of reasons why bytes 8 through 10, which together identify the reference, are only used in the final stages of the production. The first is that, while a display tells the operator which references to load into the conveyor, this monitor and its program are not connected to the PROFIBUS network. This impedes the reference from being stored immediately at the start of the line.

However, the substitution of this display for one integrated in the PROFIBUS would be worth the effort if not for other complications. After all, if a housing was associated with a reference at the start of the line, the identification vision station would be less necessary, or even obsolete. The problem arises from the fact that it is very likely that a housing will switch pallets after being machined. A drying station can have as many as four empty pallets at any given time, and operators are often responsible for more than one drying station. Therefore it becomes hard to make sure the housing returns to the same pallet. This renders writing the reference onto a tag at the start of the line worthless, and gives rise to the need of the identification vision station.

An ideal production process would take these problems into account. The current system does not keep track of individual parts and it is a clear flaw in the housings production line. One major solution would be expanding the conveyor to include an exit at every machining center. Better organization and/or visual indicators could also make sure the operators always return the housings to the their initial pallet.

Unfortunately, due to limited contact with the shop-floor, this dissertation did not try to find a solution to these problems. Instead some considerations are made in order to develop an RFID system that is fully operational by itself. While it tries to solve Renault's traceability issues when it comes to tracking individual parts, it does so without getting mired in some of the current limitations of the production line.

2.9 Nonconformities

Nonconformities in gearbox housings are defects that threaten the correct functioning of the part. Renault CACIA divides the nonconformities into raw products and machined products nonconformities. The former are largely caused by deficient casting from the suppliers. The latter are caused by deficient machining from Renault CACIA itself.

The observed raw products nonconformities are:

- **Bubbles**: occur when the gases present in the liquid metal are not released during solidification.

- **Excess or lack of material**: happen due to deficient solidification, due to a wrongly dimensioned mold riser.

- **Fissures**: occur due to inappropriate length or quality of cooling.

- **Material dragging**: happens when the mold is removed with the metal still not completely solidified, causing the material to be dragged
- **Encrustations**: when protuberances are present in the housing's surfaces, potentially causing damage to the machining tools.
- **Corrosion**: caused by exposure to water, particularly immediately after the casting process.

Machined products noncorfimities can be:

- **Deviated measures**: when the housing is machined in incorrect places, such as a bore placed incorrectly.
- **Raw product traces**: happens when the machining process is not complete.
- **Material dragging**: can happen if a tool is damaged.
- **Impacts**: can occur during the washing process or when the part is loaded into the conveyor.
- **Doubly machined characteristics**: caused by the failure of the machining center's poka-yokes.
- **Large diameters**: caused by deficient machining.
- **Tool failure**: in case a tool breaks, the housing's machining features are put at risk.

A raw product's nonconformities are visually inspected by an operator when it is loaded into the conveyor. The nonconform parts are put in a container. Later the container will be sent to the Supplier's Quality Services, a section the QA department, which will inspect the parts. Depending on the severity of the defect, the part can return to the production line or not. For example, bubbles under a certain diameter are acceptable. Raw products that do not pass the Service's tests are considered junk. The same Services also handle the nonconform machined gearbox housings. Like raw products, the machined products can return to the production line if their quality is assured despite their defects.

The other major nonconformity not referenced so far are leaks. These can be both the result of deficient casting or machining. It is important to note that a housing can pass the leak test and then be rejected due to machined nonconformities. The leak test assures the housing is leak-proof, but several of the machined nonconformities can cause structural problems, which the leak tester does not identify. Unlike with the remaining nonconformities, under no circumstance do housings with leaks return to the line.

Machining nonconformities can cause the department to trace back the part's machining center and stop the machine to analyze the cause of the problem. Currently, to trace the origin of a defect, Renault uses paper traceability sheets and the markings the operators wrote in the part during the machining stage.

Chapter 3

State of the art

3.1 Supply chains

Supply chains are networks of facilities and distribution options, whose objective is to serve a intermediate or finished product to its costumers [13]. A supply chain's elements are those who directly and indirectly are tasked with fulfilling the market's needs, and include manufacturer and suppliers, but also transporters, warehouses, retailers, and customers [14]. Elements within a supply chain see each other as partners instead of competitors, and have the common goal of increasing the competitiveness and profitability of the network. The management of a supply chain also involves marketing, product development, finance and costumer service.

3.1.1 Supply chain elements

Supply chains can have multiple degrees of complexity. For example, several different producers can be interlinked, and there is often an ultimate supplier and several subsequent suppliers. Regardless of complexity, a supply chain can be distilled to five basic elements:

- **Producers**
- **Distributors**
- **Retailers**
- **Costumers**
- **Service Providers**

A supply chain is organized as shown in figure 3.2. The costumer's request moves across the network until it reaches the producers, while the product itself physically moves from the producers to the final costumer.

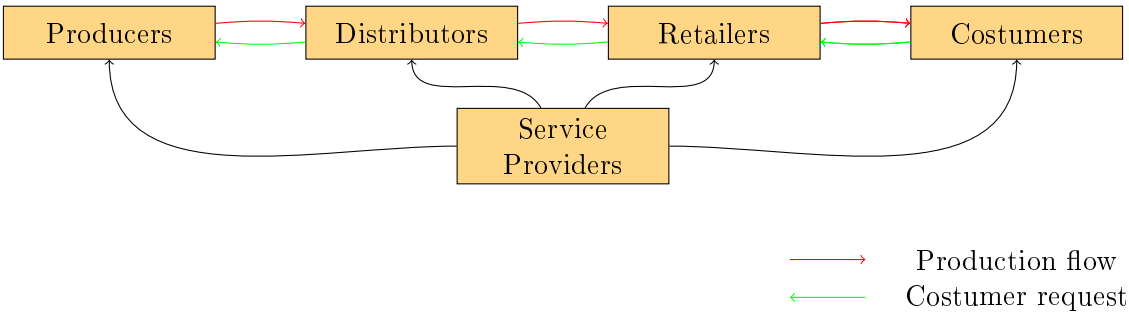


Figure 3.1: Supply chain participants, adapted from [14]

The tendency has been to increase specialization among elements of the supply chain. Each company evaluates its core competencies, deciding whether or not it is more profitable to outsource certain operations. To that end, service providers are used. These entities can provide services for different participants of the supply chain. Costumer service is an example of such service providers, servicing both the costumers and retailers.

3.1.1.1 Producers

Producers include companies or facilities that create a product. The product can be a raw material, or a finished product created down the supply chain from said raw materials. Producers are traditionally the only participants of the supply chain that add value to a product.

3.1.1.2 Distributors

Distributors are the link between producers and retailers. They buy products in bulk from the producers, buying in larger quantities than their clients would buy. This necessitates stockpiling, but simultaneously reduces the supply chain's risk of stock rupture. The result is a chain more resistant to demand fluctuations. Stockpiling also allows for quick deliveries. The other roles of a distributor depend heavily on how invested it is in the products it distributes and the resources it has access to. In addition to sales, distributors can also perform product promotion, product transportation and costumer service.

3.1.1.3 Retailers

Retailers, also called Points-of-sale, sell products to the final costumers, stockpiling in smaller quantities than distributors. They actively track the preferences and demands of their costumers, planning and advertising accordingly.

3.1.1.4 Costumers

A costumer is the final client of a supply chain, and is the main driver of the supply chain. It is the costumer's requests that fuel the supply chain's product movement. A supply chain's costumer doesn't necessarily coincide with the product's final consumer: it can also be a producer that will add value to the product, being part of another supply chain.

3.1.1.5 Service Providers

Service providers specialize in operations that the other supply chain participants deem suitable to outsource. One role commonly associated with service providers is that of warehousing and transportation. Such companies own trucks and public warehouses, and are known as logistic providers. Service providers can also supply legal services, financial services, market research and advertising, management advice, among others.

The rise of information technologies allowed companies specialized in this area to be contracted by other companies. The necessity of traceability systems and other information systems allows certain companies to develop general solutions that are adapted to specific problems within each facility or supply chain. Atena is a service provider for Renault CACIA in the area of traceability, among other areas. Section 3.10 presents some companies that work as service providers in traceability and information systems.

3.1.2 Supply chain drivers

Five areas affect the success of a supply chain. A company or set of companies hoping to improve their performance must gauge the effectiveness and needs in these areas. These areas are:

- **Production**
- **Inventory**
- **Location**
- **Transportation**
- **Information**

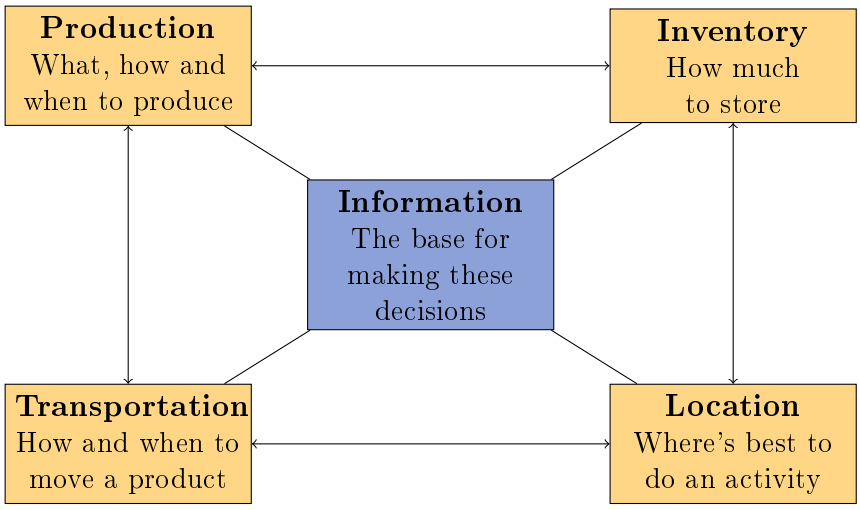


Figure 3.2: The five supply chain drivers, adapted from [14]

3.1.2.1 Production

Production is the ability of a supply chain to produce and store items. A balance must be achieved by production: high capacity factories can store high quantities of products,

responding quickly to fluctuations in demand; conversely, high production capacity is expensive, with stored products not generating any profit. Production facilities can have two focuses: product and functional focus. A facility with a product focus performs all the operations necessary to create a product. A facility with an operation focus can create multiple parts or products, but specializes in certain operations such as assembly. Both focuses are not mutually exclusive for a given facility, and can coexist.

3.1.2.2 Inventory

Inventory encompasses everything from raw materials to finished products. The inventory area faces similar problems to the production area: storing high amounts of products has its benefits and hindrances. There are three major types of inventory: cycle inventory, which is the amount of goods necessary to satisfy the needs of the costumers; safety inventory, which exceeds cycle inventory, in an effort to prevent fluctuations in demand; seasonal inventory, which is preemptively stored in seasons of the year when demand peaks.

3.1.2.3 Location

Location handles the physical location of the supply chain's facilities. It also includes the definition of which operations are carried in each facility. Location decisions will have to ascertain whenever it is more profitable to centralize operations, producing economies of scale, or when it is best to decentralize, with the intent of increasing the supply's responsiveness to shifts in demand. These decisions are based on production, warehousing and transportation costs, among other aspects.

3.1.2.4 Transportation

Transportation handles the movement of goods and materials between facilities. There are several modes of transportation: ship, rail, airplane, trucks, and in the case of some products, pipelines or electronic transport. Once again there is a trade-off to consider between responsiveness and efficiency. Ship transport is very efficient but comparatively very slow. Trucks are highly flexible while having good efficiency and relatively low costs, which makes it the most popular mode of transport for goods.

3.1.2.5 Information

Information refers to the ability of companies to make decisions based on the data gathered from the present and past state of the supply chain. Given that the information is accurate and readily available, this area can massively improve the profitability of a supply chain by allowing it to: coordinate daily activities; predict demand patterns; issue corrective measures on its operations, such as reducing lead times by better training its laborers or by better maintaining equipment.

3.2 Traceability

Traceability is the ability to trace the history, application or location of an object . A complete traceability allows to know these parameters for a single product, from its

inception (as a raw product) to retail. Traceability systems are adopted by companies because they allow companies to plan production, and aid in tracking parts associated with a defect and allow to track parts in case of a recall. A fast response to flaws in their products and processes saves money to companies, and allows to make timely corrections to their operations [15].

Traditionally traceability systems rely on paper records and manual labor. Technological advancements have shifted traceability methods towards computer databases, alongside Automatic Identification and Data Capture (AIDC) technologies, such as barcodes and RFID. The widespread use of these technologies, as well as shared information systems, allows multiple partners to access the same data across a supply chain.

3.3 Importance of traceability

The implementation of traceability systems brings several benefits to the industry as a whole. These benefits are [15; 16]:

- Stocks can be reliably monitored in real time, allowing to plan production more efficiently.
- The acquisition and expedition of products can be optimized, to the point of implementing Just-in-Time manufacturing.
- Timely detection of defects in products and rapid identification of the cause. Products can be removed from the supply chain before suffering further processes or sold to the intended client.
- Large quantities of data can be processed to deliver precise statistics to be used in reports.
- Transparency of the supply chain increases consumer's confidence in the product.

3.4 Industry 4.0

The concept of Industry 4.0 was first introduced in Germany as a roadmap for manufacturing, with the objective of achieving digitization of the the industry [17]. This fourth industrial revolution is intended to take advantage of the technologies developed over the last decades to improve the industry's effectiveness and efficiency. These technologies include RFID, IoT (Internet of Things) and cloud-based manufacturing [18].

Industry 4.0's features include flexibility in production, enabled by the interoperability between devices and systems. Interoperability is defined as the ability of two systems to communicate between each other, and to use functionalities of one another. Elevated flexibility also predicts the ability to highly customize products in small batches or even individually. Lead times are also reduced by generalized use of AIDC technologies, alongside other devices that can communicate between each other through the Internet, allowing further automation of processes. Another defining feature of Industry 4.0 is the reduction of manufacturing costs, by direct consequence of the remaining features of this revolution.

The conceptualization, discussion and application of this revolution involves several concepts, some of which are [1; 19]:

- **Internet of Things**: refers to all devices that can connect with each other via the Internet, using it to exchange data. Traditionally these devices have been computers, while nowadays advanced sensors and microprocessors are part of the IoT. Allows IoT-enabled manufacturing, which consists of using smart manufacturing objects to interact with each other, with the objective of using this information to adapt the operations they carry out.

- **Cyber-physical systems (CPS)**: systems that integrate computation, networking and physical processes. At its core a cyber-physical systems uses embedded computers, alongside networks, to monitor and act upon physical processes. Algorithms running on the computer react to data from the process's sensors and control the process accordingly. CPS are essential to Industry 4.0 manufacturing, allowing production lines to be flexible and highly automated.

- **Cloud manufacturing**: transforms manufacturing resources into services. Technologies such as RFID and barcodes can be used to digitize these resources, with the intent of using this information to provide services shared across the supply chain. It's an expansion of the concept of enterprise information systems, providing manufacturing services to all types of end users. Data stored in the cloud can extend to the whole life cycle of a product, including design, manufacturing and maintenance.

- **Smart factory**: a factory that uses its sensors, actuators and autonomous systems to be intelligent and flexible. Processes can be improved by self-optimization and autonomous decision making. Databases keep detailed records that can be used as inputs for machine learning. In essence, a smart factory implements the concepts of cyber-physical systems, IoT-enabled manufacturing and cloud manufacturing extensively. These factories can also be called intelligent or factories of the future.

- **Smart product**: products that have sensors and microchips with the ability to communicate with each other and their users, while also storing data related to their use. A key technology for the implementation of smart products is RFID, further explored in section 3.8.

3.5 Barcodes

Barcodes are one major AIDC technology. Barcode technology needs two different components: the barcode itself, which contains the data that identifies a product, and the scanner. There are two major categories of barcodes:

- **One-Dimensional (1D)** - consists of several vertical black bars of varying width, which encode the data meant to be scanned. 1D barcodes can contain up to 48 characters (on GS1-128 barcodes) [20].
- **Two-Dimensional (2D)** - these codes, usually Data Matrix and QR (Quick Response) Codes, hold the highest amount of information. The number of characters codified on a QR Code varies with its size, but can go up to 7000 [21].

The most widely used barcode is the EAN (European Article Number), created in 1976 for the grocery industry. Its precursor was the UPC (Universal Product Code), introduced in the USA in 1973. Nowadays the two systems are fully compatible, with the two types usually referred as EAN/UPC.

Barcode scanners are divided into three types [22]:

- **Laser** - The most widely used scanner for 1D barcodes. It uses laser light to read the barcode. The disposition of the black bars on a barcode codifies different characters. The scanner's laser is reflected from the black and white bars. A sensor on the scanner detects the reflected light, which is decoded by the scanner.

- **Linear Imager** - Only able to read 1D barcodes. They take a picture of the barcode and process it to decode the information within. They are better than laser scanners at reading damaged barcodes.

- **2D Imager** - Function similarly to linear imagers, but are also capable of reading 2D codes such as Data Matrix and QR Codes. On top of this, these scanners are able to read codes in any direction, which improves its scanning speed. All these capabilities make them the most robust type of scanners. A variant of 2D scanners can be used on smartphones, where its camera photographs the code and a dedicated application decodes its meaning.

Figure 3.3: Linear imager (a) and 2D Imager (b) scanners [22] [23]

3.5.1 Introduction to GS1 standard

GS1 is a standardization organization, founded in 1974, initially called European Article Number Association. It provides norms to the retail, healthcare and transport industries regarding the information methods used to manage a supply chain. Initially only applying rules to the barcode systems, it later expanded to include several other traceability tools and methods, such as RFID technologies, standardized by EPCglobal.

3.5.2 GS1 Identification keys

One defining feature of the GS1 system is the application of identification keys. Identification keys are sets of digits that identify products, documents, physical locations and more [24]. They are standard and globally unique, allowing for easy traceability along a

supply chain. GS1 defines multiple keys, namely:

Global Trade Item Number (GTIN) - Identifies an item. Most commonly used key, used on retail and supply chains alike.

Serial Shipping Container Code (SSCC) - Identifies a logistic unit. Used for transport and storage of containers in a supply chain.

Global Location Number (GLN) - Identifies a location, such as a warehouse or a store. It can also identify entities inside a company that are responsible for buying and selling products.

There are many other keys defined by GS1. These three are particularly important in traceability throughout supply chains, allowing any trading partner to know the history of a product.

3.5.3 Types of barcodes

GS1 system supports several types of barcodes. They vary in their commercial use and the amount of data stored.

3.5.3.1 EAN/UPC family

EAN codes are widely used on retail points-of-sale (POS). These codes can only encode numeric characters (0 to 9) They are divided into EAN-8 and EAN-13, holding 8 and 13 characters of data, respectively. EAN-8 is suited for small products that cannot accommodate an EAN-13 barcode. These barcodes use almost exclusively the GTIN identification key.

A EAN-13 barcode, using a GTIN key, contains four components:

- GS1 prefix, assigned by GS1 Global office. It identifies the GS1 Organization the manufacturer has joined. GS1 Portugal's prefix is 560.
- Manufacturer code - variable in length, assigned by GS1 to the object's manufacturer.
- Product code - variable in length. Alongside the manufacturer code provides a unique item reference to a product.
- Check digit - The last digit is calculated from all other digits using an error detecting code. It ensures a code is correctly scanned.

Figure 3.4: Example of an EAN-13 [24]

3.5.3.2 ITF-14

ITF-14 symbols were created with the intent of being printed and read from corrugated cardboard and fiberboard, and as such, are not used on retail checkouts. The printing

process is less demanding, and the pattern is more easily decoded by scanners. The symbol is may be used to code the GTIN when there is no need to print additional information like the product's best before date, net weight or serial number [24]. Like EAN/UPC family, they can only encode numeric characters.

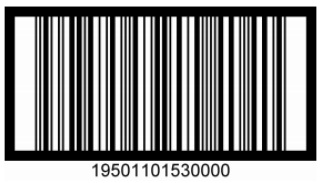


Figure 3.5: Example of an ITF-14 barcode [24]

3.5.3.3 GS1-128 symbol

GS1-128 symbology is exclusively licensed by GS1, within the Code 128 barcode. It is capable of encoding all the first 128 ASCII characters, making it an alphanumeric code. Like ITF-14, it's not used on used on retail checkouts. This type of barcode is used for products that require more than the GTIN. Some products, such as food items, may require best before date and net weight data printed on the barcode. Some other products need to be tracked individually, requiring a serial number.

Figure 3.6: Example of a GS1-128 barcode [24]

The example on figure 3.6 contains three distinct sets of digits: GTIN, best before date and a batch number. The numbers between brackets indicate the Aplication Identifier. These identifiers define the meaning of the subsequent encoded data elements, and are universal and mandatory across several barcodes, including GS1-128, Data Matrix and QR Codes. The major advantage of GS1-128 symbols is the fact they can adapt to virtually any product, allowing to encode a large number of alphanumeric characters.

3.5.3.4 Data Matrix

Data Matrix is a 2D code that can hold as much information as a 1D code in a much smaller space. A Data Matrix symbol contains an alignment pattern, consisting of an L shape in the bottom left corner of the code; and a clock pattern, which consists of two dotted lines, as can be seen on figure 3.7. When identified by a scanner, these patterns allow it to read the code in any direction.

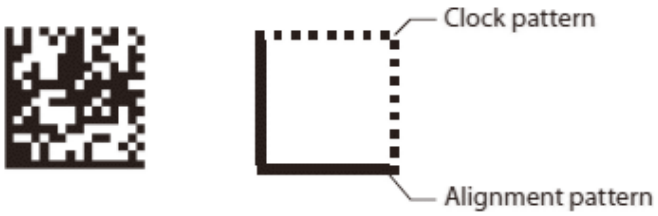


Figure 3.7: Data Matrix patterns , adapted from [25]

Each character is coded in Data Matrix by seven modules, as can be seen in figure 3.8. Each module can have a value of one or zero (black or white).

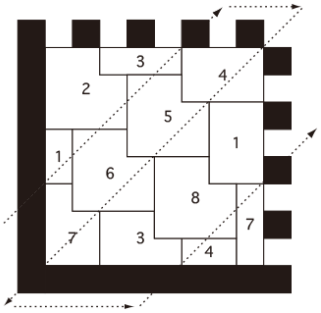


Figure 3.8: Data Matrix character arrangement [25]

Each group of seven modules represents a binary number with seven digits. The resulting number is subtracted by 1. This final number is the decimal equivalent to a ASCII character. Evidently, the more characters a Data Matrix holds, the bigger the size of the symbol. Data Matrix has a maximum data capacity of 2335 alphanumeric characters, which corresponds to a maximum size of 144x144 modules [26].

Another advantage Data Matrix has over 1D codes is that a majority of its information is used as an error correction code called Reed-Solomon code. This allows data to be recovered when a part of the symbol has been damaged. This method is used on several information formats, including CD's and DVD's, and most 2D codes.

3.5.3.5 QR Codes

Similarly to Data Matrix codes, QR codes can also be read in any direction, due to the existence of three position detection patterns, as well as timing patterns. A QR code can also have an alignment pattern, a smaller square similar to a position detection pattern, which aids position detection when displacement of modules happens.

Figure 3.9: Timing and detection patterns on QR codes, adapted from [27]

The data arrangement in QR codes is significantly different than in Data Matrix. Firstly, each character is defined by eight modules instead of seven. This allows to codify a broader range of characters, including japanese characters (kanji). In figure 3.10 the data arrangement is exemplified. The data itself is represented in gray, and like in Data Matrix codes, a large part of the information is used as error correction. Furthermore, a mask is applied to the code to prevent the appearance of the same pattern used in position detection.

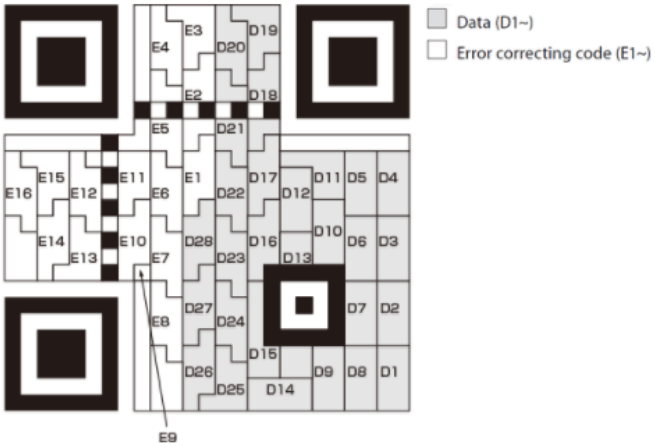


Figure 3.10: QR code character arrangement [27]

Smaller versions of QR codes are also available. The code shown in figure 3.10 is a generic Model 2 code. Model 1 codes are smaller and don't have an alignment pattern, and Micro QR only contain one position detection pattern, allowing it to have a reduced size, adequate to being printed in small components such as circuit boards. The same model can have several versions. Model 2 has 40 versions, each adding 4 modules on axis X and Y. Version 40 contains 177x177 modules, containing 4296 alphanumerical characters [28].

QR codes have been gaining popularity in recent years. Their higher information capacity and visual appearance are major appeals for commercial applications. Lastly, they are resistant to damage due to the use of the Reed-Solomon error correction method, which allows damaged codes such as those in figure 3.11 to still be accurately read.

Figure 3.11: QR code damage examples [27]

3.6 Laser Marking

Barcodes are traditionally printed on paper tags. This method is very cheap, but is also prone to failure, as tags can be damaged irreversibly in multiple ways. Laser marking provides a viable alternative to paper labels. Laser marking consists of engraving a code on a surface. The marking is created by the interaction between the surface and the laser, with the heat from the beam causing a chemical alteration on the surface's material. The technology is capable of marking on metals, like steel and aluminum, as well as plastics.

Figure 3.12: AREX 50 fiber laser marker [29]

The marks are resistant to most conditions a part can suffer. Laser marking is used to engrave alphanumeric serial numbers or other identifiers. Due to its precision, it is used to also engrave barcodes, both 1D and 2D. Fast marking lasers, such as fiber lasers, can perform their task in a matter of seconds, and have been replacing YAG and CO2 lasers in industry due to their faster performance and lower maintenance costs[30].

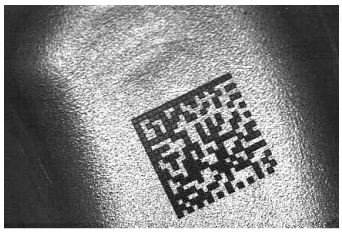


Figure 3.13: Laser marking of Data Matrix code, adapted from [31]

3.7 Dot peen marking

Dot peen marking, also called pin marking, is the process of using a pneumatic pin to mark dots on a surface. The markings are the result of the deformation of the part's material. The marking is precise and doesn't create significant stress in the material.

The dots created by the pin are closely spaced, allowing the marking of near continuous lines to imprint numbers or letters into the surface. Data Matrix codes are also marked into products using this method. Each black module can be created by a dot, or be the result of several dots together. In both cases, the end result is usually a round module, due to the shape of the pin, which can cause some scanners to not recognize the created symbol.

Similarly to laser marking, dot peen marking can mark through coats and films. The air pressure can be adjusted to change the depth of marking, and some materials may have to suffer some treatments to improve the readability of the code. While not as precise as laser marking, dot peen marking offers good quality/price ratio [30].

Figure 3.14: Dot peen marking with variable spacing between dots [32]

3.8 RFID

3.8.1 RFID system components

RFID technology allows the communication between two distinct devices, readers (also called transceivers) and tags. A reader performs requests that are sent to a tag. The tag can store information, from a simple identification number to large amounts of data that the reader intends to access [33]. Therefore this technology is prime for use in traceability, where each individual tag can be used to store data related to a product and be read through a shop floor or even a the supply chain. Information about such as a product's current location, the hour and date it passed through key locations and the processes it suffered, can be stored and accessed by multiple elements of the supply chain.

Both the RFID reader and tag (also called transponder) are equipped with an antenna, responsible for the communication with the other device's antenna. In most systems the communication starts with the RFID reader's antenna sending radio frequency waves to an interrogation zone. This is the effective zone where communication between the two devices can occur. Both the tag and reader contain microprocessors which handle communication between the devices' different elements, such as the antenna and memory [34]. The tag's microprocessor will also interpret the requests sent by the RFID reader.

The requests the reader sends are of the type read and write. These order the microprocessor to access the tag's memory, and either write or read in a position specified by the RFID reader. Read operations cause the transponder to relay data back to the reader. Write operations are usually followed by read operations to guarantee the data was in fact saved in the intended memory positions.

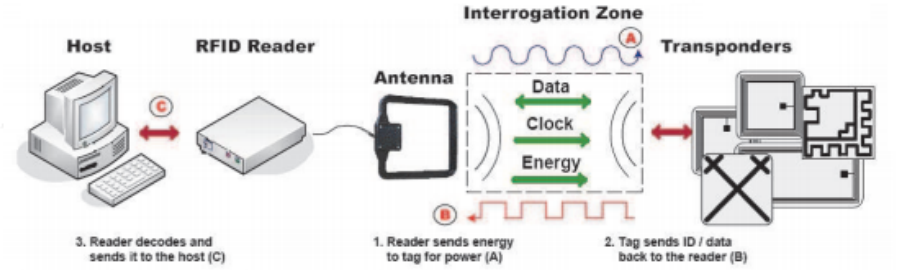


Figure 3.15: Example of an RFID system [35]

3.8.2 Types of tags

RFID tags differ in the way their integrated circuit (IC) receives power, and as such are categorized as [34]:

-Passive
-Active
-Semi-passive

3.8.2.1 Passive RFID tags

Passive tags are constituted by an antenna and a microchip, usually contained in an IC. Passive tags have no own power source. Instead they wait for a signal from the RFID reader, in the form of RF waves. These are received by the antenna and transformed into electric current that will power the IC. The microchip will then generate a response that will be read by the RFID reader's own antenna.

Due to the reduced amount of components, passive tags are both cheap and durable, compared to other types of RFID tags [36]. They are generally smaller than active tags, which suits them to be mounted on products of reduced size.

Within passive tags also exist chipless tags, also called smart labels. These do away with chips, decreasing their production costs, in an effort to compete with very cheap and widespread barcodes [37]. Chipless tags store less information, with some only storing an ID of the product. Surface Acoustic Wave tags are one type of chipless tags. These are printable tags, whose piezoeletric substrate has printed wave reflectors. The board has a Interdigital Transducer that receives a scanning pulse from an RFID reader. The transducer transforms this pulse into an acoustic wave that is propagated across the board. The signals reflected form a unique pattern that constitutes the data contained within the tag.

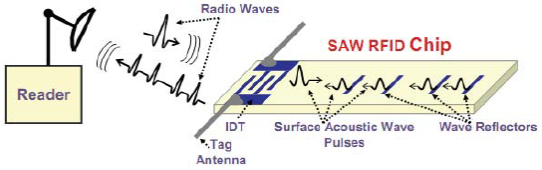


Figure 3.16: Passive SAW RFID tag, adapted from [37]

3.8.2.2 Active RFID tags

Active RFID tags contain, in addition to an antenna and an IC, a power supply, normally a battery or even a solar cell [34]. In these tags the IC is powered with the voltage from the battery. Active tags can be divided into transponders and transmitters. Transponders act much in the same way as passive tags, in the sense that the reader is the first to communicate, searching for tags. The existence of a power supply results in the need of a far weaker electromagnetic field produced by the RFID reader to achieve the same result as a passive tag, since in passive tags a large quantity of the field is used to power the IC. One direct consequence of this is the extension of the communication range between tag and reader.

Transmitters, instead, transmit data every few seconds to any reader in its range. Transmitters are in general more energy consuming, depending of course on how often they are programmed to transmit data and the strength of the signal.

The inclusion of a power source increases the size of active tags when compared to their passive counterparts. This also increases its price. For these reasons, active tags are used to track bigger, more valuable products, such as cargo containers, large components or machinery.

3.8.2.3 Semi-passive RFID tags

Semi-passive tags also have a power source that feeds the IC. However, it does not provide enough power for the antenna to generate a high frequency signal. For that the tag still needs the electromagnetic field from the RFID reader. These tags cannot act as transmitters, and have less communication range than active tags, but higher than passive tags. In sum, they offer a middle term between active and passive tags.

3.8.3 Communication methods

The three main communication methods between the reader and the transponder are the following [34]:

-Inductive coupling
-Close coupling
-Backscatter coupling

3.8.3.1 Inductive coupling

Inductive coupling functions in a principle similar to a transformer. Both the reader and transponder have an antenna coil. The reader's coil generates a high frequency electromagnetic field. If the tag's coil is in range, this field will pass through the coil's cross section, generating a voltage that, after being rectified, will power the microchip. As such, inductive coupling is mostly associated with passive tags.

The communication range between devices using inductive coupling ranges from a few centimeters to at most 1 meter [38]. The 13.56 MHz frequency is the most associated with this communication method, though it is also used in low frequencies.

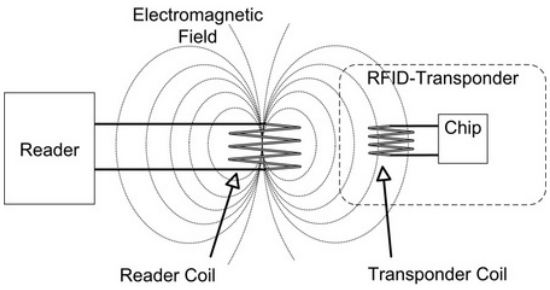


Figure 3.17: Inductive coupling [39]

3.8.3.2 Close coupling

Close coupling also functions similarly to a transformer. However it is far more limiting than inductive coupling, being distinct due to the necessity of physical contact between the transponder and the reader. A typical reader has a ferrite core with two coils, which are powered by the reader's voltage. When a card is inserted in the air gap between the core, a current is induced in the card's coil.

Communication ranges using close coupling are no bigger than 0.1 to 1 cm. This method is mostly used with security cards.

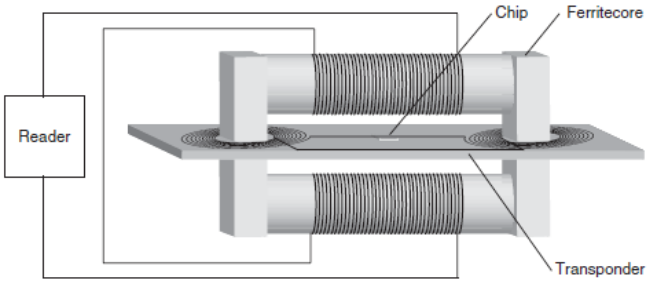


Figure 3.18: Close coupling [34]

3.8.3.3 Backscatter coupling

Backscatter coupling functions using electromagnetic waves, and can also be called electromagnetic backscatter. The reader's antenna emits an electromagnetic field. When that fields reaches the tag, some of it will be used to power the tag's IC, if the tag is passive. A fraction of the field will be reflected back and ultimately be picked up by the reader. The reflected signal can be modulated switching on and off a resistor parallel to the transponders antenna (modulated backscatter). This effectively alters the reflection cross section, and the resulting pattern is interpreted as data by the reader.

This method is associated with UHF. The use of such high frequencies allows the transponder's antenna to be much smaller than if frequencies lower than 30MHz were used .

3.8.4 Frequencies used in RFID

RFID systems use a variety of frequencies to communicate. Presently, the four categories of frequencies are [34; 36]:

-Low Frequency (LF)
-High Frequency (HF)
-Ultra High Frequency (UHF)
-Microwaves

3.8.4.1 Low Frequency (LF)

These frequencies range from 30 to 300 kHz. The most used are 125 and 134 kHz, with 134 kHz being one of the first widespread frequencies. LF is associated with close coupling and inductive coupling. Therefore its communication range is limited, although it remains a very good solution for metallic environments, with its low frequencies hardly being disturbed by interference. Other applications include the control of accesses, animal tagging and ticketing.

3.8.4.2 High Frequency (HF)

While a HF can be anywhere between 3 MHz and 30 MHz, in practice, mainly 13.56 MHz is used. It is associated with inductive coupling and passive tags, and for that reason is ideal for use in supply chains to control the flow of products or containers. Furthermore, it is one of the few frequencies that is regulated and used around the world. Its shortcomings are poor range and being prone to interference, particularly in metallic environments.

3.8.4.3 Ultra high Frequency (UHF)

UHF range from 300 MHz to 1 GHz and are used on both active and passive tags. For active systems frequencies of 433 and 915 MHz are common. Passive tags use frequencies between 800 and 900 MHz. These frequencies are typical of backscatter coupling, and therefore are used for far greater ranges than the two previous groups of frequencies. One problem stifling the widespread of UHF is the fact different countries and regions around the world have regulations in place that only allow certain frequencies. The result can be seen on figure 3.19 This heterogeneity impedes tags read with UHF in China to be read in Europe. Efforts have to be made globally to allow a certain UHF to be nearly universal, much like as 13.56 MHz is for HF.

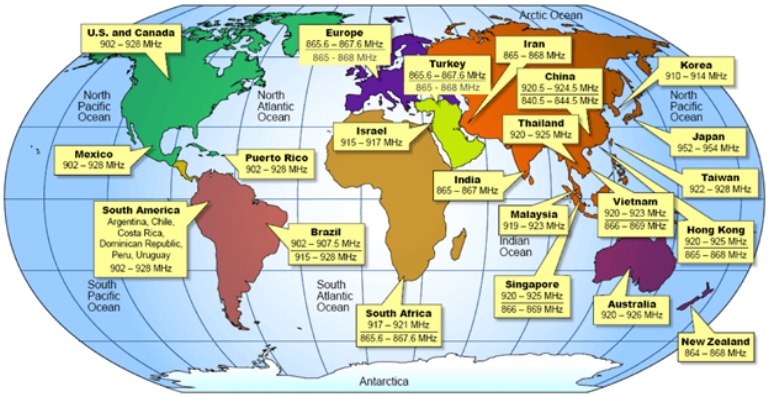


Figure 3.19: UHF allowed in different countries/regions [40]

3.8.4.4 Microwaves

Waves with frequencies higher than 1GHz. 2.4GHz frequencies are the most used and regulated around the world. Function similarly to UHF, being associated with backscatter coupling, with similar communication ranges. Examples of applications of microwaves are tracking large items and automatic road tolls.

3.8.5 EPCglobal

EPCglobal was started with the intent of providing standards for the use of RFID tags. It is managed by GS1 and intends to provide the framework to replace barcodes in the future. To that end it is an open standard that any company can join. In an ideal scenario, companies around the world would have their product's data tracked using the EPCglobal standard, allowing the traceability of every product throughout its supply chain.

3.8.5.1 EPC

EPCs are Eletronic Product Codes, unique keys that identify an individual object. Since GS1 keys are widely implemented, EPC was designed to have correspondence with several GS1 keys, while improving the older system in order to allow the identification of unique products. As an example, the Serialized Global Trade Item Number (SGTIN) is a EPC scheme akin to the GTIN key. However it expands on it by adding a serial number.

The SGTIN syntax is the following:

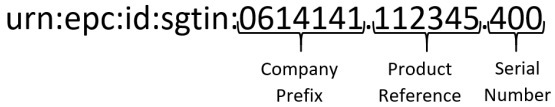


Figure 3.20: EPC schema for SGTIN

The first set of numbers are the GS1 company prefix, attributed by the GS1 local management to a certain company (the prefix includes GS1 prefix and the manufacturer

code). This is the same identifier used in GTIN. The second set of numbers is the product reference, which identifies a class of products. The product reference present on the SGTIN is not a direct correspondence to the one present in the GTIN, since EPCglobal's intent is to be a general norm, while GTIN can have varying sizes, from GTIN-8 to GTIN-13. The last three digits correspond to the serial number, which is assigned by the manufacturer, allowing to trace products individually. Unlike the other two sets of numbers, GS1 does not constrain the length of the serial number, which be assigned by the manufacturer at will.

Several other EPC's directly correlate with GS1 identification keys, such as Serial Shipping Container Code (SSCC) and the Global Location Number (GLN). Each conversion from one GS1 keys to EPC has its specific rules, including concatenating strings or rearranging the position of identifiers.

Any tag adhering to EPCglobal must follow this schema for the SGTIN, including the alphabetic characters, which serve to identify the EPC scheme.

3.8.5.2 GS1 EPCglobal Network

EPCglobal Network is a set of technologies and standards that allow data from different companies to be exchanged easily. One such standard are EPC schemes, mentioned previously. Others are:

- **EPC tag** - tags whose data is arranged according to the EPC norms.

- **EPC reader** - reads the EPC tags.

- **EPC Middleware** - A software that integrates the processing of data received on the RFID reader with the information services (EPCIS), executing the communication between the systems. It can also manage data transfers with other services such as ERP (Enterprise Resource Planing).

- **EPC Information Services (EPCIS)** - A standard platform that allows multiple supply chain partners to access information. The information is managed by the EPCIS, and can be stored on EPC repositories, which each company can search through with a global interface.

- **Discovery Services** - Allows authenticated authorized clients to access additional information about a product. While companies that only use EPCIS have access merely to non-serialized products, companies allowed to use Discovery Services can track serialized products through the value chain. This is the ultimate objective of the EPCglobal Network.

Figure 3.21 shows a representation of this very same system, illustrating how two companies who use all the technologies and standards covered by EPCglobal can have knowledge of the history and current location of any given object.

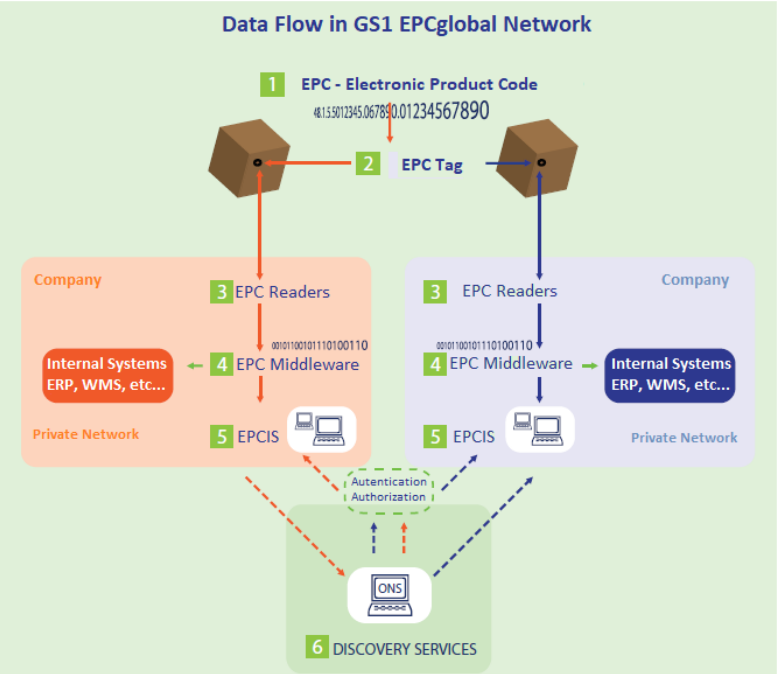


Figure 3.21: EPCglobal Network representation, adapted from [41]

3.8.5.3 Pros and cons of RFID

The adoption of RFID systems is trending upward around the world because of several reasons, which are [1; 33; 36; 42]:

- Allow to store dynamic data in the tags, which can be added or updated at any station with an RFID reader.
- Possibility of tracing products through supply chains.
- Possibility of using data contained in the tag to automate the processes a product goes through.
- Flexibility in the reading of RFID tags, with most systems not requiring visual contact between the reader and the tag.
- Tags have high memory capacity, compared with other methods such as barcodes.
- Many tags are resistant to extreme manufacturing conditions, such as the presence of lubricants or high temperatures.
- RFID standards employ anti-colision algorithms, which allows to read multiple tags at once.

Nonetheless, RFID system have disadvantages that stifle its spread:

- Acquisition costs are still higher than printing barcodes. The expenses are even higher for tags designed to operate in extreme conditions.
- The selection of the RFID hardware must be careful, since some communications are put at risk in metallic environments.
- Difficulty in implementing information systems that meet the needs of the company, all the while taking full benefit of the memory capabilities of tags.

- Existence of multiple frequencies. Especially problematic with UHF, as can be seen on figure 3.19.

- Lack of normalization of RFID communications. Even the same frequency can have multiple standards, such as ISO14333A and ISO15693 for 13.56MHz. Readers and tags using different standards cannot communicate with each other.

The high costs of RFID tags, paired with the fact that different companies across the supply chain will have different RFID transceivers, operating at different frequencies and standards, lead to most companies using RFID tags on a closed loop. In closed loops the tags are associated with products at the start of a facility and are released from their product at the end. The tag then returns to the start of the process, while the product is shipped to the next facility.

3.9 Solutions proposed by others

3.9.1 "Integração de Sistemas de Rastreabilidade em Ambiente Industrial", Universidade de Aveiro [43]

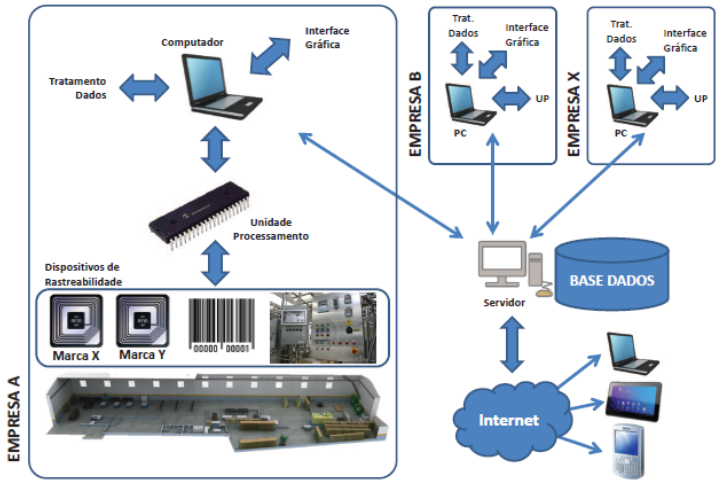


Figure 3.22: Traceability solution proposed by Bártolo [43]

As seen in the figure 3.22, inside the same company can coexist multiple traceability devices. This problem can occur throughout a supply chain as well. The solution proposed in this dissertation tries to minimize these problems, associating an RFID tag at the start of the production line, while maintaining the tag the product arrived with. Each station has two RFID readers of two different standards, capable of communicating with the two distinct tags. A microcontroller is connected to all traceability devices. It receives commands from the computer, verifies the structure of the message and stores the received bytes in a message that will be in accordance with the standard of the specific device the message is intended to. The microcontroller also processes the tag's responses and relays them to the computer. The computer will treat this data and present it to the user on a graphical interface. It also communicates with a server that allows access

to a database. This database in turn permits other companies to view this information.

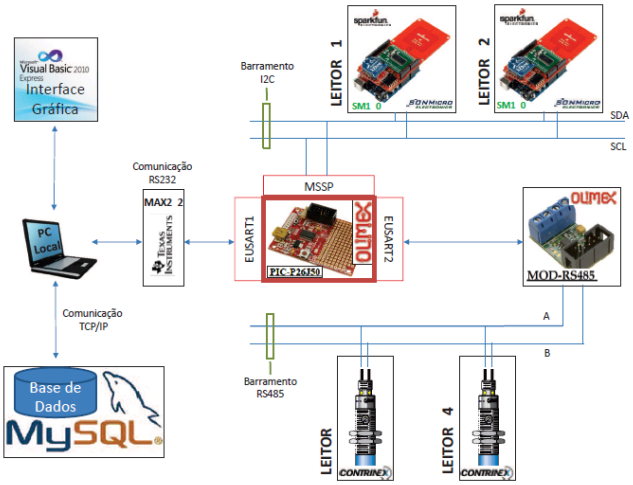


Figure 3.23: Implementation proposed by Bártolo [43]

The implementation integrates two types of RFID readers. The Sparkfun SM130 readers communicate with the microcontroller through I2C-bus, being compatible with the ISO 14443A standard. The Contrinex® readers communicate through RS485, supporting the ISO15693 standard. The company itself utilizes ISO 15693 tags, transferring data from the tag already on the product to the new tag. This new tag is utilized throughout the line until it reaches the shipping area. Here, once again, two readers are utilized, and relevant data is transferred to the tag that arrived with the product.

The implemented database has two types of data: static and dynamic. The ID's of objects, companies and readers are independent from the movements and operations of objects, and so they are considered as static data. Dynamic data includes: information about the arrival and departure of objects; production registers; product aggregation between each individual part and their pallet or container; and tag association, allowing to know, from the object's ID, the two tags it is associated with.

3.9.2 "Novo Sistema De Rastreabilidade Industrial"(2012), Universidade de Aveiro

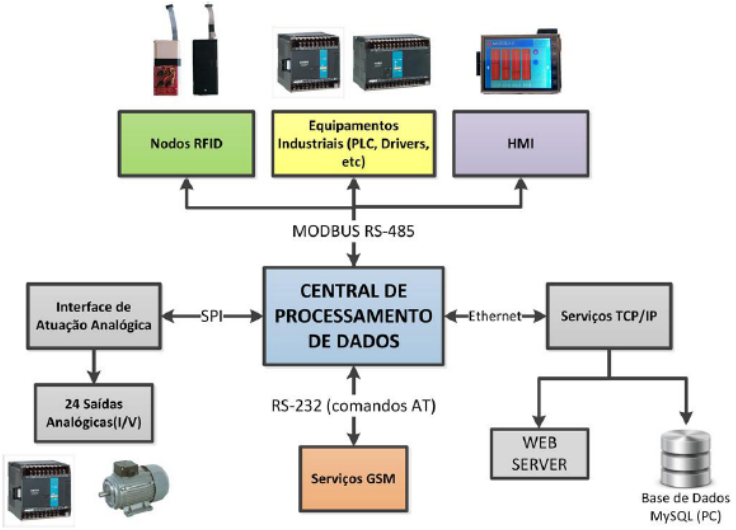


Figure 3.24: Implementation proposed by João Almeida [44]

The solution proposed in this dissertation is a centralized architecture with a microcontroller as the central processing unit. The microcontroller is connected by MODBUS RS485 to the factory's equipments, such as PLCs, HMI consoles and RFID readers. Communication with RFID readers uses MODBUS enhanced, a variant that allows the integration of RFID technology. 24 analog outputs are also used, allowing for a more versatile solution. In that vein the solution also uses GSM services, allowing SMS text messages, and TCP/IP services to communicate with a server and a MySQL database.

The implemented system is constituted by a central module, handling data processing, and external modules. The central module contains a microcontroller integrated in a printed circuit board (PCB). This board already contains the hardware support to integrate GSM and TCP/IP services. GSM services are used to warn a production supervisor in the event of some anomaly or a particular operation, using for that effect SMS messages. The web services use a open access Microchip library to perform real-time monitoring of production lines and provide information about the traceability services. In this implementation the web services also controls the 24 analog outputs.

The automatic identification devices, namely RFID readers, are part of the external modules. SM130 readers were used, using the ISO 14443 standard. The microcontroller constantly asks information from the readers. The readers are constantly searching for an RFID TAG. When one is found, the TAG is first authenticated. Afterwards the tag's 64 blocks of 16 bytes are ready to be written or read. A HMI was also implemented to control and monitor the 24 analog outputs.

3.9.3 "A Cyber-physical System Architecture in Shop Floor for Intelligent Manufacturing" [45]

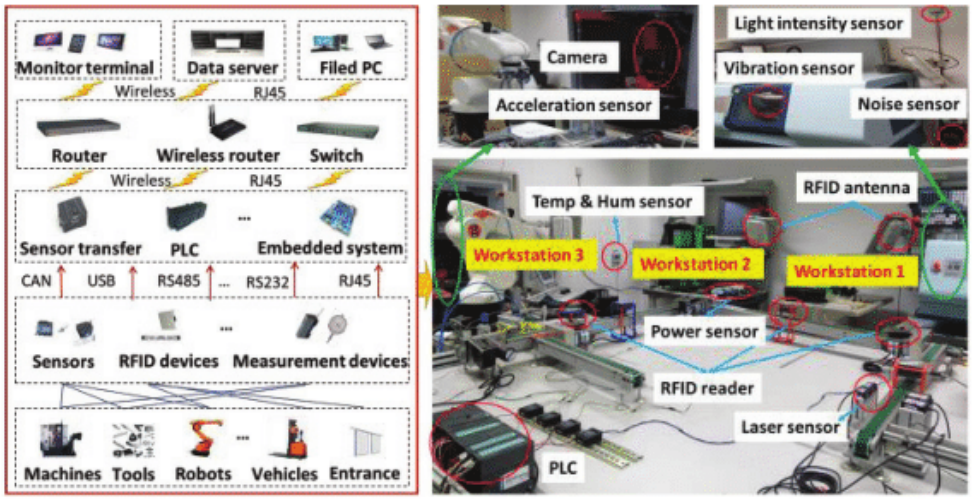


Figure 3.25: Solution proposed in Liu et al.

This article written by Liu proposes a cyber-physical system to be used for intelligent manufacturing. This system has three layers. The first is a physical connection layer, consisting of several sensors that gather data about the production line in real time. The middleware layer aims to transfer the data received from the first layer to the central server, while also receiving commands from a computation layer and acting on the line's controllers. The third layer is the computation layer. The computation layer uses information from the line's sensors or from Enterprise Information Systems and runs them through algorithms and specific models. The intent is to extract underlying patterns from machine working conditions and manufacturing processes, which would be used to act upon the line's machines for operation control and maintenance.

The implementation was performed on three workstations: workstation 1 is a CNC micro lathe, workstation 2 is a CNC milling machine and the third consists of a CNC milling machine in addition to a robot. Several sensors and RFID readers are associated with each workstation. Each sensor and each station has its own specific middleware, which in general consists of an embedded system integrating Java web and an Arduino platform. These devices communicate through the Internet to the computation layer. Data transferred to this layer is organized in a unified JSON format, with each message having the same fields, only changing the content of each field depending on the source of the data (sensors and RFID readers). The implementation also specifies the use of monitor terminals, which include computers and mobile phones, to visualize information about the line in real time.

3.9.4 "Real-time information capturing and integration framework of the Internet of manufacturing things" [46]

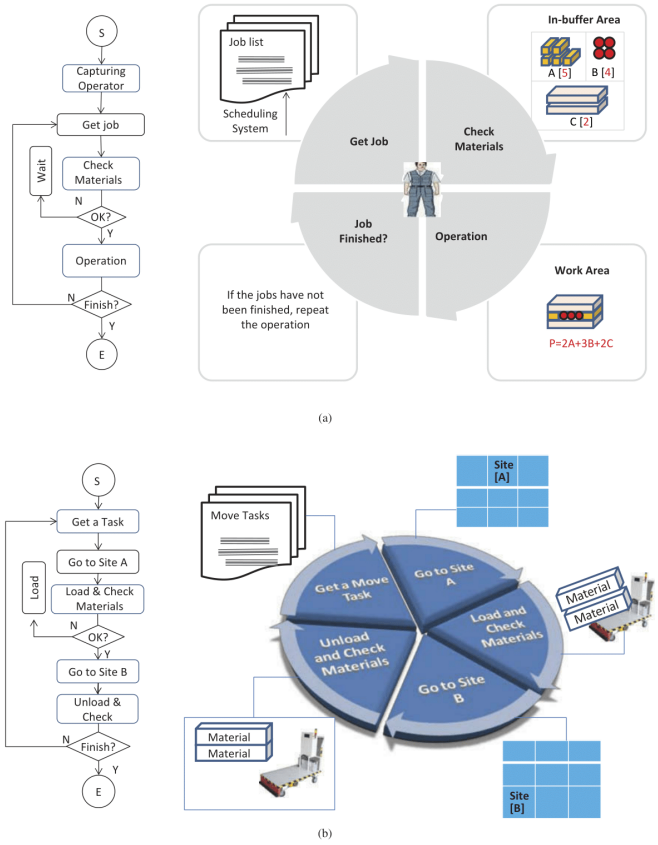


Figure 3.26: Solution proposed in Zhang et al.

This article by Zhang et al. proposes a IoT solution for manufacturing. It uses a specific line, namely an Internet of Manufacturing Things (IoMT) lab for applying its solution. The solution consists of a physical smart object layer that uses mostly RFID tags and transceivers. The objects communicate with a middle layer that performs data processing, transforming raw data from the readers into standard information.

The proof of concept shop floor is composed by three workstations for producing and assembling products, each also containing a robot arm to load and unload parts. A storage area is also present in the shop floor, and an automated guided vehicle (AGV) is used to transport items between the storage and the manufacturing areas. A shop-floor management system is also used to plan manufacturing and controlling the production.

Figure 3.26 shows the work flow from two different perspectives. The top part shows it from the perspective of the operator at a given workstation. The worker receives a job from the manufacturing planning. He starts by registering his presence with his unique RFID tag. Next he uses the same reader in the machine to check that every material corresponds to the bill of materials (BOM) for this job. He will do a sequence of operations until the job is concluded. After this the operator goes to do the next job.

The AGV does a loop between the storage and the manufacturing area. Much like

the operator, it too receives a task from the up-level management system. It will go to the first site on its path and get loaded with the necessary materials or parts. The AGV contains a UHF reader with considerable range which identifies each object it is loaded with. The AGV will count the materials until its load fits the order it received, and then carries to the second site, where its cargo will be unloaded. The AGV will also count unloaded items, being aware of when it is ready for another move task.

3.9.5 "A case of implementing RFID-based real-time shop-floor material management for household electrical appliance manufacturers" [47]

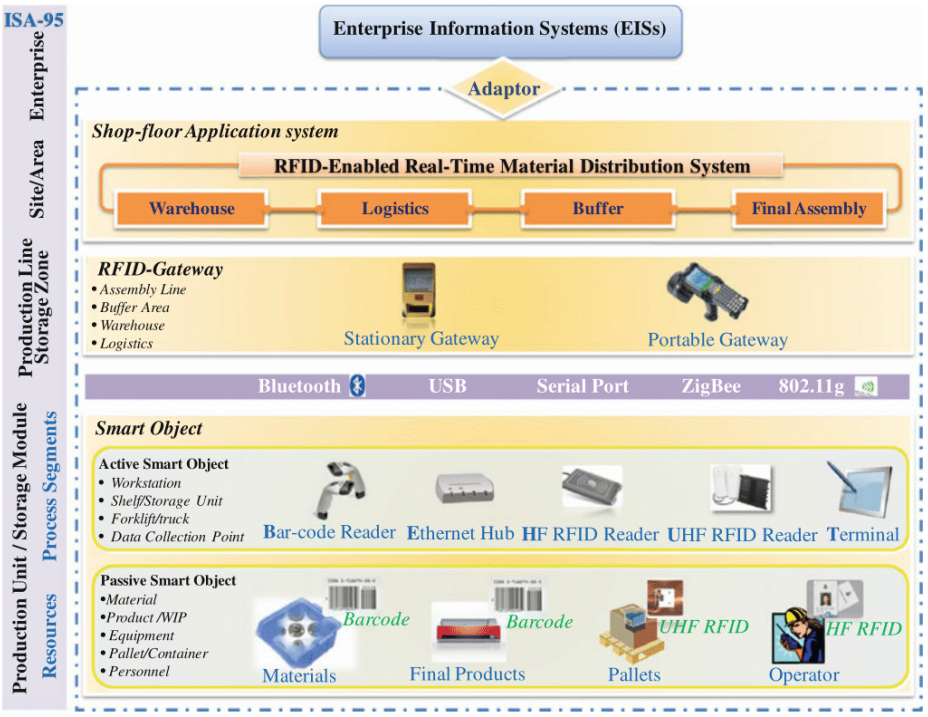


Figure 3.27: Solution proposed in Qu et al.

This article presents an RFID based solution with its focus on material management. The company it applies its solution to is an air conditioner manufacturer in China. The factory specializes in the assembly of final products, buying materials and parts from several suppliers and subsidiaries. The case study focuses on one workshop, divided into a production department and a logistical department. The logistical department stores materials temporarily before entering the assembly, which is performed in the production department. Each workshop is supplied by the factory's warehouses, while the production planning department manages production and material requests.

The solution presented in the article consists of the use of passive tags in tracked objects, which are read by active smart objects such as barcode and RFID readers. The association of tags follows a closed loop application, wherein RFID tags are associated with objects that are moved within a cycle and eventually return to a starting point. Such objects can be pallets and boxes, while individual materials are associated with

paper barcodes. The active objects communicate with an intermediate layer composed of RFID-Gateways, which are consoles equipped with hardware capable of communicating through USB, Serial Port, Bluetooth and Wi-Fi, as well as drivers corresponding to several smart objects. Furthermore, RFID-Gateways use standard web service interfaces, used to communicate with the already implemented ERP systems, which allows planning department's and warehouse orders to communicate with the workshop's gateways. The gateways can also be either stationary or portable, with this article giving a major emphasis on portable gateways.

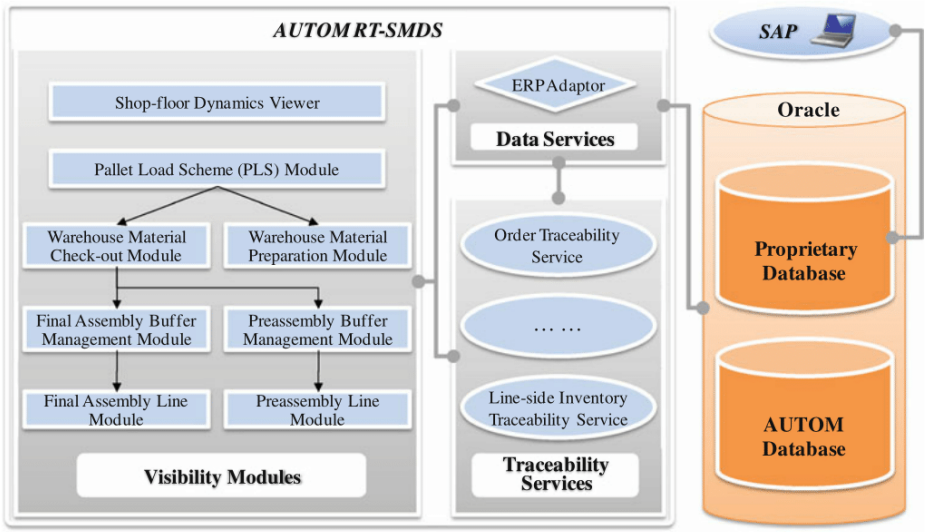


Figure 3.28: Architecture of implementation proposed in Qu et al.

The implementation of this system is based on a service-oriented architecture (SOA), with the intent of providing a real-time shop floor material distribution system (RT-SDMS). The system consists of visibility modules located in several locations. They communicate with the traceability services of the ERP through web pages. The implemented system provides support to the manufacturing process, from the production planning to the final assembly. The manufacturing process is composed by following stages:

Step 1 - Production manager makes production and material requirement plans. Manager reviews several data: status of current orders, status of production lines and material availability.

Step 2 - The logistics manager creates a pallet loading scheme (PLS) whenever a new material order arrives.

Step 3 - Material preparation in warehouse. All PLS are visible in the warehouse's gateways. The materials have barcode tags that are associated in the gateway to the RFID in the pallet. Finished pallets are sent to the shipping dock.

Step 4 - Logistics manager assigns material picking task. The corresponding buffer gate's gateways informs the logistics operator.

Step 5 - Logistics operator picks the pallet, confirming with his gateway if he is unloading the correct order.

Step 6 - Warehouse keeper ensures materials in the pallet correspond with PLS.

Step 7 - Preassembly material buffering. Some orders will go to a preassembly, with some other going directly to the final assembly. When the production line needs to be replenished it sends a request to the buffer area gateways.

Step 8 - Associate preassembled with the pallet tag. Deliver to the final assembly buffer.

Step 9 - Deliver both materials and preassembled parts to the final assembly line whenever they are requested.

Step 10 - Final product assembling, packaging and warehousing. Products are tagged by barcodes and associated with their pallet tag. The materials used in the production are deducted from the inventory. Logistics managers created tasks to pick up the finished products and transport them to a warehouse.

3.10 Commercial solutions

3.10.1 Balluff

Balluff is a company invested in developing automation technologies for Industry 4.0. They develop and sell solutions for industrial sensors, computer vision, human machine interfaces and RFID. Their RFID technologies are ready to be integrated in multiple ways. One possible use for their RFID tags is asset tracking, by keeping track of detailed information of objects such as machine tools and molds. The primary information held on RFID tags on these products are set-up parameters, tool life data , part matching and off-set parameters [48]. Off-set parameters are gathered by a pre-setter, a device composed of several cameras to identify a tool. The use of RFID tags to track machining tools is particularly useful. Since the same machining center can hold multiple tools for a variety of operations, keeping track of how many times a tool is used is a challenge with conventional methods.

Another use of their products is Electronic Kanban. Kanban itself is defined as a system used to reduce cycle times, using a philosophy of Just-in-time to pull the production from the demand towards the supply. Its electronic variant replaces traditional Kanban cards with RFID tags, further reducing cycle times, adding traceability capacities to the Kanban system and further automating the process. The use of certain products sends messages to previous stages of the production line, also reducing inventories. These signals can even be sent across the supply line, increasing inventory flow.

The company's solutions also assist on Production Control in Assembly Automation. The RFID tags present in products can serve to store Build Information. This type of information is used on flexible manufacturing and assembly lines. The tag contains the data necessary to inform the line equipment what operations are associated with a particular component. Alternatively, the tag can contain only a unique identifier to a database entry, which contains all the information pertaining to the product. Lineage Information can also be contained within the tags, and allows the traceability of individual parts, documenting each part with unique identifiers. This type of information is necessary to trace back the source of a defect.

Finally, the company also predicts the cases where a company only needs to do interplant logistics. In this case only containers or large pallets containing multiple products are tracked with RFID tags. This is a similar solution to the one employed by Renault

CACIA, which uses, however, GALIA tags. RFID tags permit, however, their reuse, storing higher quantities of information and a quicker, more reliable traceability system.

Balluff sells RFID systems of LF, HF and UHF. The architecture of their installations can be dedicated or expandable, the latter allowing to install multiple RFID readers as the company's needs rise. Their HF products are compliant with ISO 15693 and ISO 14443.

3.10.2 GlobeRanger

GlobeRanger specializes on traceability software development. The three major markets it serves are: aviation, where it offers automated inspections of emergency equipment; police departments, employing software that keeps track of shared and permanent assets [49]; and manufacturing, where the company's software can track products through all the manufacturing process.

GlobeRanger implements a platform that uses the data read from RFID tags to trace products. This platform visually demonstrates to the user the current location and journey of a component. Alongside the use of IoT, this solution allows companies to automate decade old processes to track items, still very reliant on paper records. This replaces labor intensive, costly and error-prone tasks into digital automatic processes.

The result is the traceability of inbound and outbound products, as well as products stored in the plant and in the production line. GlobeRanger claims it achieves a 5 to 1 return on investment with the its software solutions [50]. And in general the system reduces lost products, eliminates errors in data inputting and does away with cycle counts, presenting reliable data for logistics and the financial departments.

3.10.3 Contrinex

Contrinex is a manufacturer of sensors for manufacturing automation. It produces inductive, photoelectric, ultrasonic and capacitive sensors, as well an array of RFID products. Its RFID solutions vary in their purposed use. Figure 3.29 shows three readers. The leftmost reader communicates solely through USB, and is designed for user access control stations and tag programming by PC. The second reader in the same figure is made of stainless steel, and communicates through RS485. It is especially designed to work in extreme environments, being insensitive to dirt and performing successfully in metallic environments. These transceivers communicate solely in LF, namely 31.25 kHz. Their communication standard is specific to Contrinex, therefore they only communicate with Contrinex tags of the same standard.

The third transceiver shown in figure 3.29 communicates also through RS485, but is more general purpose than the previous reader. It does not perform as well in metallic environments, partially because it communicates with HF of 13.56 MHz. This reader is compliant with the ISO/IEC 15693 standard, being able to communicate with any tag of the same frequency and standard. Contrinex readers are sent commands in order to perform operations such as searching for tags. These commands must be sent in the ContriNET protocol, which defines the structure of messages sent to and from readers. HF and LF transceivers can be present in the same bus, so long as ContriNET is used, regardless of the control interface used (some interfaces are shown in figure 3.31).

Figure 3.29: Selection of Contrinex RFID readers [51]

Some of the several Contrinex tags are shown in figure 3.30. Depending on the application, tags can be subjected to different environment conditions. The first tag on the figure shows a general purpose tag, of Contrinex's basic range of products. These are 13.56 MHz HF tags, compliant with ISO15693. They are not designed to be embedded in metal. However, Contrinex states that the presence of a nonmetallic spacer between the tag and the metal surface yields a good communication, even with readers from the basic range.

Nonetheless washdown tags such as the one shown in the middle of the picture are especially designed to be embedded in metal, partially due to the fact they communicate in 31.25 kHz. Moreover these tags are especially designed to be in contact with liquids in washing operations. They are highly corrosion-proof, saltwater resistant and withstand aggressive solvents. The last tag shown in figure 3.30 is non-embeddable and communicates in 13.56 MHz, but is resistant to high temperatures. It can be subjected to 250 °C, due to its liquid crystal polymer housing. Other Contrinex tags can only be subjected to 125 °C. Several other tags exist from these three types. They vary in their memory size and communication range.

Figure 3.30: Selection of Contrinex RFID transponders [51]

Generally the RFID readers are connected to control interfaces such as shown in figure 3.31. These interfaces serve as the middleware between the readers and a controller or host that controls the communication. As many as 32 Contrinex readers of any type can be connected simultaneously to one of these interfaces. The two interfaces shown vary in their communication with the controller. The first communicates through USB, and is the only one of this kind from Contrinex. The second interface shown communicates

with the upper layers by PROFIBUS. Several other models of this type communicate through different protocols, including Ethernet and PROFINET.

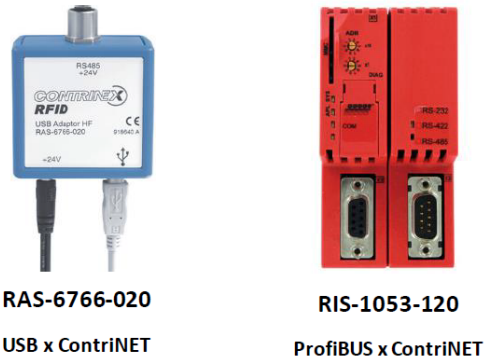


Figure 3.31: Selection of Contrinex RFID interfaces [51]

Chapter 4

Solution

4.1 Proposed solution's architecture

The proposed solution intends to introduce a cheap and flexible traceability system to a production line that uses RFID readers, based on the housing line of Renault CACIA as a specific example. One of the major components of this proposed system is the processing unit. This solution predicts the use of a cheap, low power microchip to perform this function. This piece of hardware serves as the middleware of the traceability system. It receives messages from peripheral devices, processes them, and sends the relevant information to the top layer of the system. This top layer is an Internet server, which the processing unit communicates with using Wi-Fi.

The bottom layer consists of several RFID readers. The possibility of communicating with other peripherals other than RFID readers is predicted in this solution, however the focus of the solution will be on the use of RFID to create smart objects. To achieve this, the solution considers that every pallet should contain an embedded RFID tag, as is already implemented at Renault CACIA. However this solution will expand upon this by fully using the RFID's capabilities, increasing the data stored in the tags.

The RFID readers will continually search for a tag in their field. Whenever a tag is present in the field, the reader sends a message back to the processing unit. Then it is the role of the ESP32 to process this information and subsequently send and receive all the remaining messages that entail the communication with the RFID peripherals.

At the same time, the processing unit must also use its Wi-Fi capabilities to communicate with a remote server. This server, containing several php pages, will be expecting messages from the middleware. The server can perform two tasks. It can receive data from the microprocessor, or can be used by the microprocessor to access data from two other sources: the database, and the graphical interface.

The graphical interface should have its files in the same host computer, exchanging information with the server's php pages. This graphical interface should also communicate in two ways: by displaying information gathered from other elements of the system; and by sending information, through the server, to both the database and ESP32. The information sent to the processing unit can be then stored in the RFID tags that pass through the line. Particularly, the system expects operators in each line to use the graphical interface to select which nonconformities are present in the products they are handling in their station.

The last major component of the proposed solution is the database, containing all

the data deemed necessary for an effective traceability system. The proposed database specifically applies to Renault CACIA's housings line, and is further explained in section 4.3.

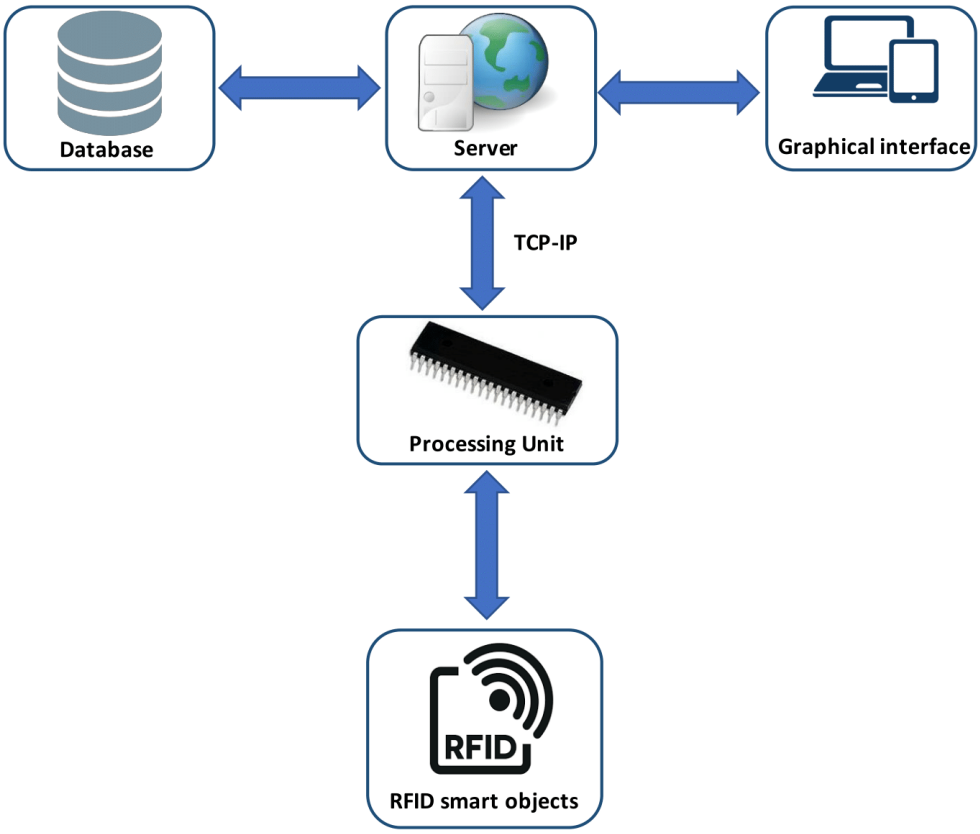


Figure 4.1: Proposed solution diagram

4.2 Data Processing

The proposed solution contains two main data processing elements: the processing unit, which communicates with the RFID readers; and the server, containing php files that make the link between several other parts of the system. Separately neither could offer the full capabilities of the solution. The processing unit cannot communicate directly with the database, always needing a Wi-Fi request. And the php files are not a physical device capable of communicating with the readers.

4.2.1 Data processing in processing unit

The readers search for RFID tags in their field repeatedly. When the reader finds a tag in its field it sends a message to the microprocessor containing the Tag's UID. This identifier is then used to send read and write commands which specify this tag as its target. The write commands will vary depending on the workstation the RWM is assigned to.

The processing unit will also determine whether the response it gets from the command is correct. In case the responses to the writing or reading commands are not what is expected, which can be caused by several situations, such as an error in the communication or the tag leaving the field prematurely, the processing unit will return to the first mode of operation: continuously searching for a tag.

If, as expected, the tag leaves the field after all operations have been performed, the data read from the TAG will be sent through a request to a page. This page will then process the data received and send it to the database.

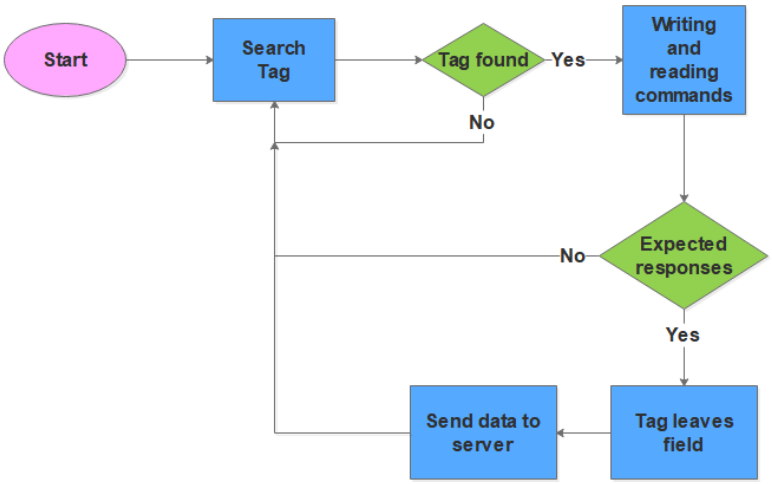


Figure 4.2: Data processing in ESP32

4.2.2 Data processing on server

The server will be the link between the database, the processing unit and the graphical interface. It will contain several php pages that are expecting requests containing data. These pages will perform several actions:

- Process data sent from the processing unit, and send it to the database. Each station in the production line has a unique number that is sent in the HTTP request. This number identifies the operations the php page is programmed to do. Data sent from each RWM is expected to be stored in certain tables of the database, which will differ from reader to reader.

- Store temporary data in database. This data will be used to show details of the line in real time.

- Process data, both temporary and permanent. This data will then be accessed by the graphical interface to update several of its elements in real time. Some of the processing can be simple, such as reading one single value or string, or more complex, like compiling larger sets of data to be represented in visual form.

4.3 Proposed database

Databases are essential tools for storing large amounts of data. A production line such as the housings line in Renault CACIA needs to retain data concerning the products it

manufactures. It is therefore imperative to create a database that is flexible, intuitive, and above all, normalized.

Several models exist to conceptualize a database, namely Entity Relationship Diagrams (ERD), Unified Modeling Language (UML) and Functional Dependency Diagrams (FDD). The latter can be developed from the previous two. FDD's can also be conceptualized directly, without needing to identify entities (or classes in the case of UML).

The proposed solution used Entity Relationship Diagrams to conceptualize the database.

4.3.1 Entity Relationship Model

ER models are used to map the structure of a database. Peter Chen first introduced this model in 1976, alongside a particular notation. The model identifies three main components: entities, relationships and attributes [52].

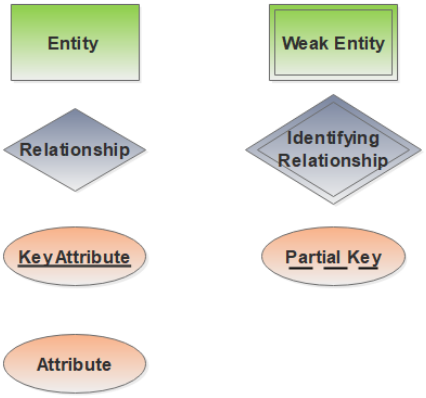


Figure 4.3: ERD notation for entities, relationships and attributes

An entity is any concept that exists either physically or logically. In the particular case of the proposed database, an entity can be a housing or the passage of a housing through an RFID reader in a particular station.

Relationships describe how entities relate with one another, usually through a verb. In the case of the housing, it 'undergoes' a passage through an RFID reader.

Attributes are associated with entities, and are data that is needed to describe or characterize a certain entity. In the case of a housing, its serial number is an attribute. In fact, it is a key attribute. A key attribute helps identifying a unique instance of an entity , and therefore, the serial number is the key attribute of the housings entity set. A entity can have multiple key attributes if necessary. The attributes needed to identify an entity form what is called the primary key.

Figure 4.3 shows the difference in representation between key attributes, partial keys and normal attributes. ERDs also distinguish between normal entities and weak entities. Weak entities are ones where none of its attributes can unequivocally identify one instance of the entity. Weak entities need key attributes from another entity to identify them uniquely. In that case, we have a identifying relationship between the two entities. Visually a weak entity is represented with a double line box. Identifying relationships are also represented with a double line 'diamond'.

Chen's notation for relationships between entities can be seen on figure 4.4:

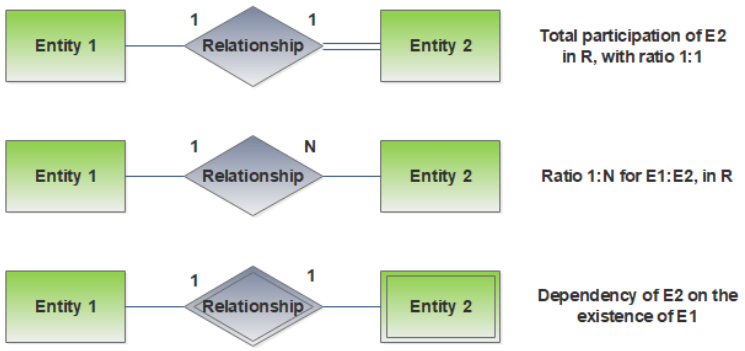


Figure 4.4: Chen's ERD notation for relationships between entities

The top relationship occurs between two regular entities. This relationship is characterized by a 1:1 ratio, meaning that every instance of Entity 1 (E1) is related with one single instance of E2. This first example also has a double line that connects E2 to the relationship. This double line represents total participation of E2 in the relationship with E1. This means that every instance of E2 corresponds to one instance of E1, while instances of E1 can be present in the database without being associated with any of E2.

The second example shows a case wherein the ratio is 1:N. Each E1 corresponds to one E2, while multiple instances of E2 can correspond to the same instance of E1. In this case both entities have only partial participation.

The last example in Figure 4.4 shows the notation used whenever one entity of a relationship is a weak entity. As previously stated, since E2 is dependent on the primary key of E1, its box is represented with a double line. Similarly, since the relationship is of the identifying type, it too is represented with a double line.

There are other symbols used in Chen's ERD's, namely for derived or multivalued attributes. However, for the database proposed in this solution, they were not necessary.

4.3.2 Database's Entity Relationship Diagram

The proposed solution will be applied to four workstations in the production line. These are the Loading Station, Machining Station, Leak Tester Station and Unloading Station. The proposed solution considers that at each of these stations exists a RFID reader. The pallets, containing the housings, will have a tag associated with it, and will pass through these four stations one time for each housing.

These four stations were chosen due to their relevance to the manufacturing process and the product's traceability. The Loading, Leak Tester and Unloading Stations all perform nonconformity tests on the housings and the results of these tests must be stored in the database. The Machining Station is also essential to the product's traceability since each machining center can be the source of nonconformities. One notable consideration made for this proposal is that an RFID reader is located at each machining center, which is not currently implemented at Renault CACIA. This would entail changes in the conveyor, since currently RFID readers are only located at the drying stations, of which there are only six, compared to the twelve machining centers. This consideration removes the production line's problem of housings changing pallets mid production. This way,

a housing identified at the start of the production will assuredly remain with the pallet containing the RFID tag.

A passage of an housing through these four stations corresponds to an entity of the production line. Several other entities were identified, alongside their relationships and attributes. Figure 4.11 shows the ERD for the proposed database, with every entity and relationship explained in sections 4.3.3 and 4.3.4 respectively.

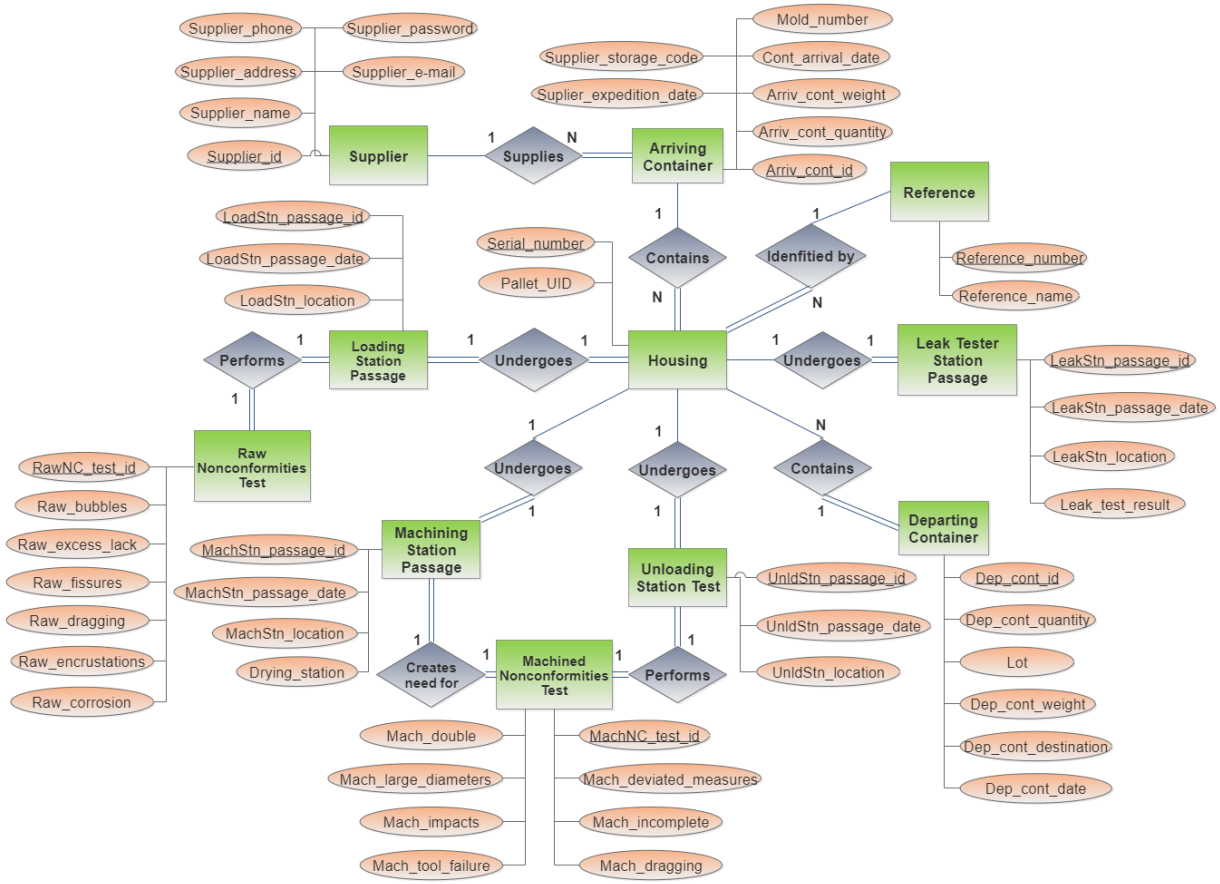


Figure 4.5: Proposed database Entity Relationship Diagram (ERD)

4.3.3 Database's Entities

The following entities were identified in the proposed solution for the database:

- **Housing**: The object that needs to be tracked throughout the production line, and the entity the database revolves around. Its key attribute is the serial number, a unique identifier for every housing. The other attribute associated with a housing is the pallet's UID, since the housing travels through the line on top of the pallet, which holds the RFID tag. This UID is the same as the tag's UID. One possibility not contemplated in this solution is considering the tag or pallet as their own entity. This would be the case if more attributes could be linked to the tag or pallet themselves. A situation such as this occurred when assigning the attributes for the housing's reference.

- **Reference**: This entity refers to the type of housing. Initially its key attribute, the reference number, and its other attribute, the reference name, were attributes of the housing entity. Upon further deliberation, it became apparent the reference should be its own entity. Firstly, its two attributes are interchangeable. This would lead to unnecessary repetition of data, mainly in the table or tables which would hold the data concerning the relationship between the housing and other entities . Secondly the entity of a single housing is different from the entity of the type of housings. Thirdly, while not present in this solution, each reference can have its own attributes that further define it, such as dimensional specifications, which necessitates the establishment of the reference as its own entity.

- **Arriving Container**: The entity regarding containers that enter the production line, either from other factories or Renault CACIA. The key attribute is the container's number, printed into the GALIA tag by the supplier. All of the remaining attributes are present in the GALIA: expedition date, the storage code of the supplier, the quantity of housings, the container's weight and the mold number.

- **Supplier**: Every container that arrives the line has a supplier. Every supplier has a unique ID as its primary key. The other attributes are information such as the name, address, phone number, e-mail and password, in case the two suppliers are given access into each others information systems.

- **Loading station passage**: This entity refers to a single passage by a housing through the station where the housing is loaded into the conveyor. The operator places the housing on top of a pallet which contains the RFID tag that will be read in this station. This passage is given an ID. The date and the location of the station are defined as attributes of this entity as well. Each production line has one loading station, therefore the passage's location can be identified merely by indicating the combination of the Atelier and production line.

- **Raw Nonconformities Test**: This entity represents the visual test performed by an operator when a housing passes through the loading station. Each test is given an ID as its primary key. The remaining attributes are results to the test. These nonconformities are explained in section 2.9

- **Machining station passage**: Represents the passage of a housing through one of the several machining centers in a line. Also identified by an ID, its attributes are the date, the machining center location and the drying station. The location in this case details not only the Atelier and line, but also the machining center the housing goes through, of which there are twelve.

- **Unloading station passage**: Similar to the remaining stations, identified by an ID. The date and location are also stored in the database.

- **Machined nonconformities test**: A test similar to the one performed in the loading station. Differs only in the nonconformities themselves, which are specific to the

result of the machining process, and can also be found on section 2.9

- **Leak station passage**: Entity similar to other station passages. Since a leak test is performed to each housing, the result of the test is associated with this entity, with an 'O' meaning the housing passed the test, and 'X' meaning it failed.

- **Departing container**: The container that is shipped from the housings production line. The container's ID is the same as the finished products GALIA. The attributes of this entity are the date the GALIA was printed, the quantity, the weight, the lot the housing is associated with, and the destination of the container.

4.3.4 Database's Relationships

This section explains how the relationships present in the diagram shown on figure 4.11. Not all relationships are explained in detail. Some relationships are similar to the following examples, being mentioned whenever the same logic can apply.

4.3.4.1 Supplier-Arriving Container-Housing

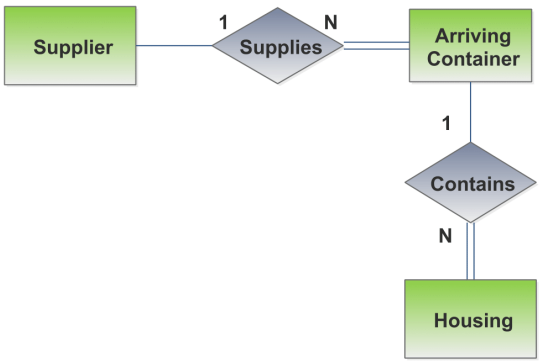


Figure 4.6: Relationships between the Supplier, Arriving Container, and Housing

The relationships between these three entities share many similarities. Noticeably, they both have a participation ratio of 1:N. For example, a single supplier can supply several containers. Similarly, one container will certainly contain multiple housings.

Another feature of these relationships is that they demand total participation from one of its entities. An arriving container has forcefully one supplier. The same happens with any given housing: it must have arrived in a certain container.

4.3.4.2 Housing-Departing Container

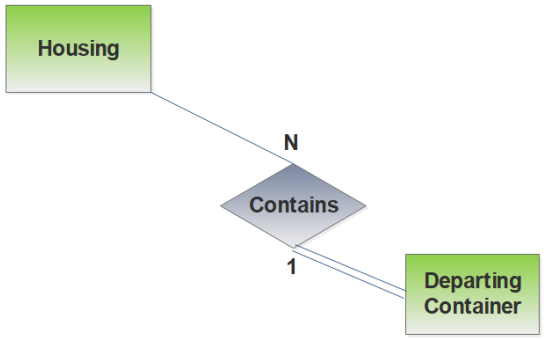


Figure 4.7: Relationship between the Housing and the Departing Container

The relationship between the housing and departing container is very similar to the one between the housing and the arriving container. The major difference rests in the fact that this relation demands the total participation of the departing container. The previous relationships had only partial participation of the arriving container because a container can be present in the database without its housings being already identified by their serial number. Meanwhile, every departing container is filled with already existing housings in the database, therefore it participates fully in the relationship.

4.3.4.3 Housing-Reference

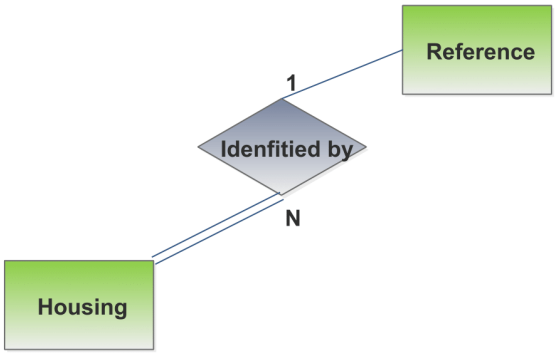


Figure 4.8: Relationship between the Housing and the Reference

This relationship is characterized by a ratio of N:1, meaning several housings can have the same reference. The housing entity has total participation, with each housing having mandatorily a reference number. Meanwhile, though unlikely, the database can have references stored that do not correspond to any housing that passed through the line.

4.3.4.4 Housing-Unloading Station Passage- Raw Nonconformities Test

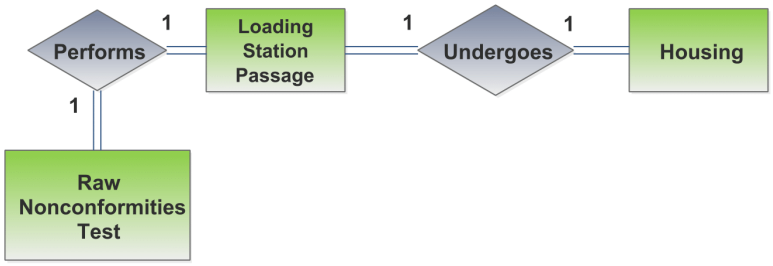


Figure 4.9: Relationships between Housing, Unloading Station Passage and the Raw Nonconformities Test

The relationships here presented have a ratio of 1:1. Any housing must have passed through the loading station only once. And any passage of a housing corresponds to only one raw nonconformities test. Regarding the total or partial participation of these entities, they all have total participation in their respective relationships. A housing must pass through the loading station, and an RFID reading in that station is only possible if a housing exists. Likewise, a housing passing in this station necessitates a nonconformities test.

The primary key of the passage through the Loading Station is a unique ID. If the primary key of this entity was instead the location of the station, this would turn the entity into a weak one. In that case the location itself could not identify the passage of a single housing: several housings go through the same station. The passage would need two primary keys: the location and the Serial Number of the housing, making it, effectively, a weak entity. This circumstance was avoided in this and all other entities: a unique ID removes the need for more than one key to define a passage, as well as removing and the existence of weak entities.

4.3.4.5 Housing-Machining Station Passage

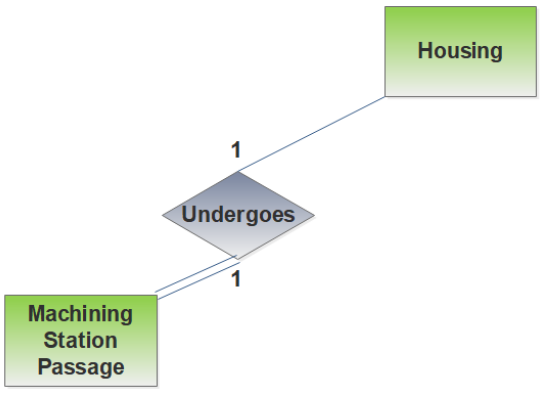


Figure 4.10: Relationship between Housing and Machining Station Passage

Contrary to the relationship between the Loading Station and the Housing, the Housing's relationship with the passages through other Stations is quite different. In the proposed

solution the housing is inserted in the database on the Loading Station, but this can also be its last passage through the production line. If nonconformities resulting from the casting are found, the housing is removed from the production line. In that case the housing remains in the database, having its own serial number, but no further relationships occur. The same happens with a departing container. A housing may never be loaded into a container shipped to assembly if it fails any of the tests throughout the line.

4.3.4.6 Machining and Unloading Stations Passages - Machined Nonconformities Test

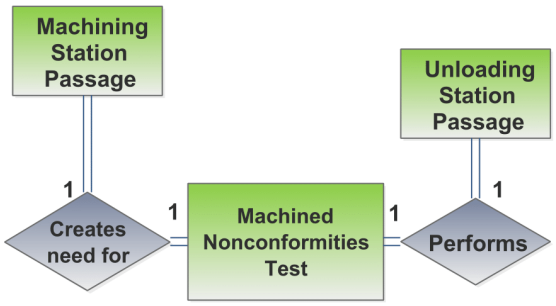


Figure 4.11: Relationships between Machining/Unloading Stations Passages and the Machined Nonconformities Test

These relationships are close to those shown in the previous section , between the Raw Nonconformities Test and the Loading Station. However, in this case two different stations are related to the nonconformities test. The nonconformities themselves are created during the machining process, but are tested only at the unloading station, where an operator performs a visual inspection. Therefore the nonconformities test has a relationship with both these two stations.

4.3.5 Database's Tables

The ERD presented in section 4.3.2 is the framework around which the database is built. Once identified all the entities, relationships and attributes, it is possible to create a normalized database. Several rules can be applied to translate the diagram into a normalized set of tables. The first, and most important, is that each relationship corresponds to at least one table. In the case of relationships with a ratio of 1:1, such as the ones between the Housing and the several passages throughout the line, two approaches are available for creating tables:

- Create a table that essentially merges several entities. Since each instance of a housing corresponds to a passage through a station, this is a valid way to construct the database. However, and especially given the number of 1:1 relationships between the Housing entity and other entities, this option would result in a table with too many columns.

- Create two tables. One table concerns one of the entities in the relationship 1:1, having this entity's key as the table's primary key. The other entity's key is used as a foreign key in this first table. The second table concerns merely the second entity, and the only key present in this second table is the second entity's primary key. In this solution, the first table refers the user to the second table through the foreign key.

For ratios of 1:N, there is merely one viable approach of creating tables. Since one of the entities can be linked with several instances of the other entity, the solution is the one presented in the previous paragraph. Except in this relationship ratio it is not ambivalent which entity's table contains the foreign key. For 1:N ratios the entity corresponding to N must hold the foreign key, which refers to a single line of the other table.

The rules would change if any entities in these relationships were weak entities. In that case, the weak entity table would need a foreign key. However, since every entity is identified by a unique ID, no entity is weak. Using the foreign key approach for 1:1 relationships, coupled with the fact the Housing entity is the N portion of several relationships, this results in the Housing table holding its own primary key (Serial_number), the entity's other attribute (Pallet_UID), plus all the remaining primary keys of other entities it has relationships with, as can be seen on the following figure, showing every table of the database.

1- Housings Table:

Serial_ number	Pallet_UID	Reference_ number	Arriv_cont_ id	LoadStn_ passage_id	MachStn_ passage_id	LeakStn_ passage_id	UnldStn_pas sage_id	Departing_ cont_id

2- Loading Station Table:

LoadStn_ passage_id	Date	Location	RawNC_test _id

3- Raw Nonconformities Test Table:

RawNC_test _id	Bubbles	Excess_lack	Fissures	Dragging	Encrusta- tions	Corrosion

4- Reference Table:

Reference_ number	Reference_ name

5- Machining Station Passage Table:

MachStn_ passage_id	Date	Location	Drying_ station	MachNC_ test_id

6- Unloading Station Passage Table:

UnldStn_ passage_id	Date	Location	MachNC_ test_id

7- Machined Nonconformities Test Table:

MachNC_ test_id	Deviated_ measures	Incomplete_ machining	Dragging	Impacts	Double_ machining	Large_ diameters	Tool_failure

8- Leak Tester Station Table:

LeakStn_ passage_id	Date	Location	Leak_test_ result

9- Arriving Container Table:

Arriv_cont_ id	Arrival_date	Expedition_ date	Quantity	Weight	Supplier_sto- rage_code	Mold_ number	Supplier_id

10- Supplier Table:

Supplier_id	Name	Address	Phone	E-mail	Password

11- Departing Container Table:

Dep_cont_id	Date	Quantity	Weight	Lot	Destination

Figure 4.12: Tables used in the database

The Housing table is the single most important table in the database. It includes the keys of seven other tables: Unloding, Machining, Leak and Unloading Station Passages, as well as Arriving and Departing Containers, and finally the housing's reference. Some of these tables also contain foreign keys of other tables. The Loading Station table

refers to the Raw Nonconformities Test table, using the second table's primary key. Both Machining and Unloading Station Passage tables contain the primary key of the Machined Nonconformities Test, with all these relationships having a 1:1 ratio. Finally, the Arriving Container Table contains the Supplier's key, in a 1:N relationship.

4.3.6 Functional dependency diagrams

Another possible tool to conceptualize a database is a functional dependency diagram (FDD). In fact, FDDs are many times the next step in conceptualizing a database after having drawn a ERD. FDDs can however be drawn without previously identifying all entities and their relationships.

A functional dependency is a relationship between two attributes wherein one attribute, say X, is associated with another attribute, Y. This relationship can be represented as X→Y. In this case, the X is called the determinant set and Y the dependent set.

After drawing a functional dependency diagram, one can use it to create the database's tables based on that diagram, following certain norms. The Boyce-Codd normal form (BCNF) is a set of rules that can be used to normalize a database. BCNF states that in a normalized database, all the table's candidate keys (equivalent to primary keys in entity relationship model) must be that table's determinants, ie. the attributes upon which all other attributes are dependent.

Already the similarities between this normalization form and the rules used for ERDs can be spotted. Figure 4.13 illustrates the Serial_number's functional dependencies, as well as the subsequent relationships that spring from the unloading station passage ID. Only a part of the proposed database's dependencies are shown, with the intent of showing how normalization through the BCNF is performed without repeating the entirety of normalization process already performed using the Entity Relationship model.

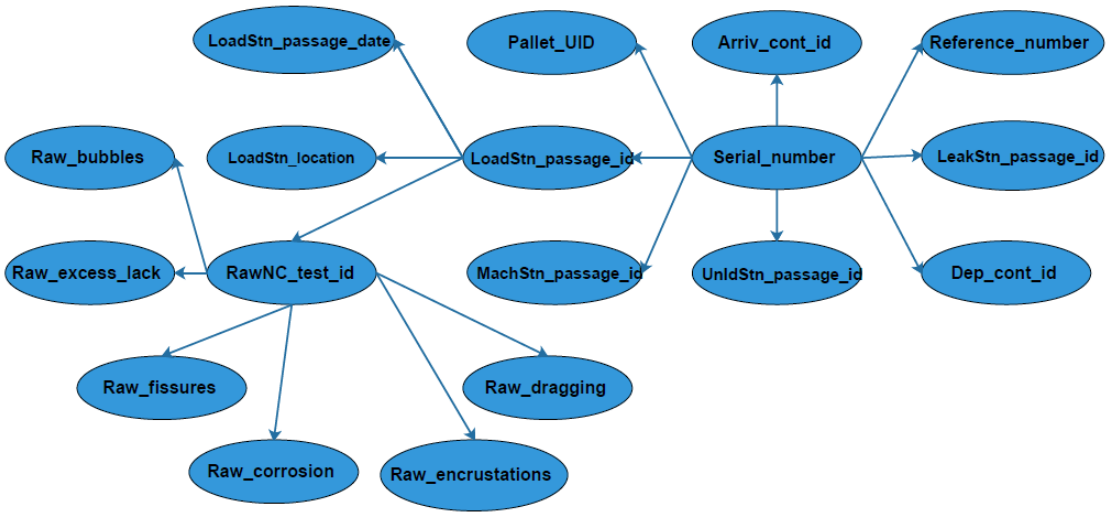


Figure 4.13: Partial functional dependency diagram of the proposed database

Following BCNF, we can identify that three different attributes are determinants in the illustrated relationships . However, as previously mentioned, BCNF states that every

determinant must be a candidate key in a table, and that each table must have only one candidate key. This is not the case in the universal relationship shown in figure 4.13, whose determinants and candidate key are shown in table 4.1.

Determinants	**Candidate keys**
Serial_number	Serial_number
LoadStn_passage_id	
RawNC_test_id	

Table 4.1: Universal relationship's determinants and candidate keys

Since three determinants are present in this initial diagram, there are a total of three sub-relationships in this universal relationship. They are identified onward as R1, R2 and R3.

R1 was defined as the relationship where the Serial_number is both the sole determinant and candidate key, as shown in figure 4.14

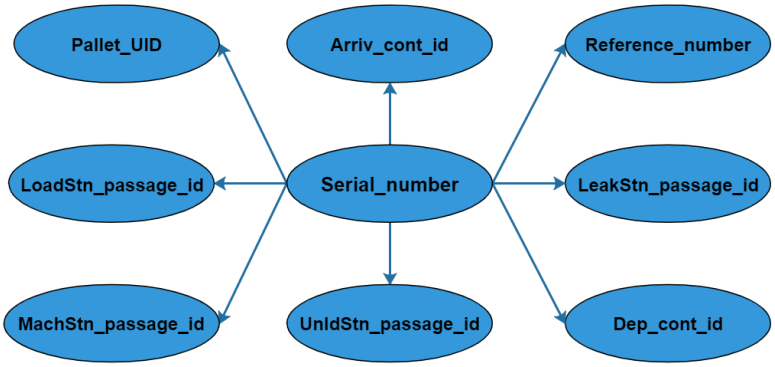


Figure 4.14: R1: Serial_number is determinant and candidate key

R2 is the relationship which has the UnldPost_passage_id as the determinant of all attributes. One of those attributes is the RawNC_test_id, which itself is the determinant for several other attributes not present in the R2 sub-relationship.

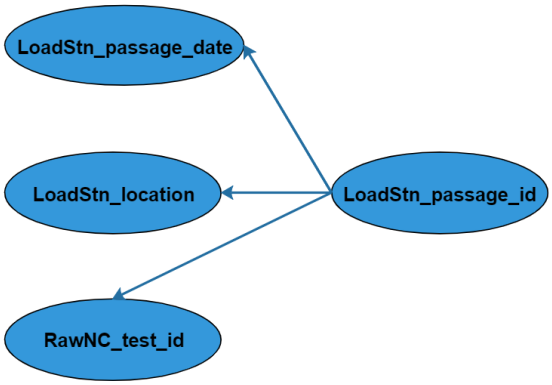


Figure 4.15: R2: UnldPost_passage_id is determinant and candidate key

Finally, the last sub-relationship identified in the universal relationship presented in figure 4.13 has RawNC_test_id as determinant and candidate key. This relationship, called R3, is represented in the following figure.

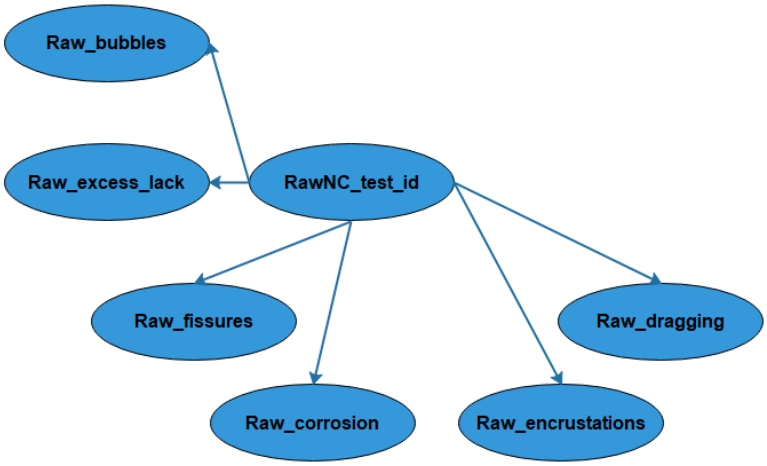


Figure 4.16: R3: RawNC_test_id is determinant and candidate key

Arrived at this stage, the relationships have been normalized, and each of these sub-relationships matches a table, in accordance with BCNF. The determinant and candidate key of the relationships is the primary key of said tables. The tables would then be the following:

R1:

Serial_ number	Pallet_UID	Reference_ number	Arriv_cont_ id	LoadStn_ passage_id	MachStn_ passage_id	LeakStn_ passage_id	UnldStn_ passage_id	Departing_ cont_id

R2:

LoadStn_ passage_id	Date	Location	RawNC_test _id

R3:

RawNC_ test_id	Bubbles	Excess_lack	Fissures	Dragging	Encrusta- tions	Corrosion

Figure 4.17: Some of the tables created by using BCNF

These tables, R1 to R3 perfectly match the tables pertaining to the Housing, Loading Station and Raw Nonconformities Test entities presented in figure 4.12. While only three entities and their relationships were normalized utilizing BNFC, the same would happen to the remaining entities, as their attributes follow the same pattern. Every remaining entity is identified by a unique ID that determines the other attributes, as well as determining the unique ID of other entities it has relationships with.

It is important to note that, while the resulting tables are the same, there were other options for the tables created from the ERD. For example, it was decided that the Housings Table would contain the foreign keys of every other entity it has relationship to. This

coincides with the normalization according to BNFC. However, for 1:1 ratio relationships it was valid that the Serial_number would be the foreign key in the other entity's table, such as, for example, the Loading Station Table. This discrepancy happens because ERD does not distinguish between determinant and dependent sets for 1:1 relationships. Therefore it can be concluded that BCNF is a more restrictive normal form than the rules used in Chen's model. This happens even though the database's tables are exactly the same, but only due to considerations made in ERD normalization.

Chapter 5

Implementation

5.1 Implementation architecture

The implemented architecture consists of three main layers. The bottom layer contains all the RFID readers, located at different stations of the housings production line. As figure 5.19 shows, these are the Loading Station, the Machining Station, the Leak Test Station and the Unloading Station. The passage of the pallets and their tags through these stations allows data to be read and written into the tag's memory. The Contrinex read/write modules (RWMs) selected for this implementation (detailed in section 5.2.1) communicate with an ESP32 microchip (section 5.2.3), which is integrated in an IoT embedded system developed for this implementation.

The ESP32 uses its inbuilt Wi-Fi antenna to communicate with a server hosted on a remote computer. The server's php pages process the information received from the middleware and update the database, developed in MySQL. The server also processes data so it can be shown in the graphical interface, itself implemented through web pages, which serve as services for the line's operators and managers. These pages were developed with Bootstrap, a free front-end framework for web-development, and Highcharts, a javascript library built specifically for dynamic graphs. These will be explained further in section 5.7. Figure 5.19 shows a diagram with the implementation's architecture.

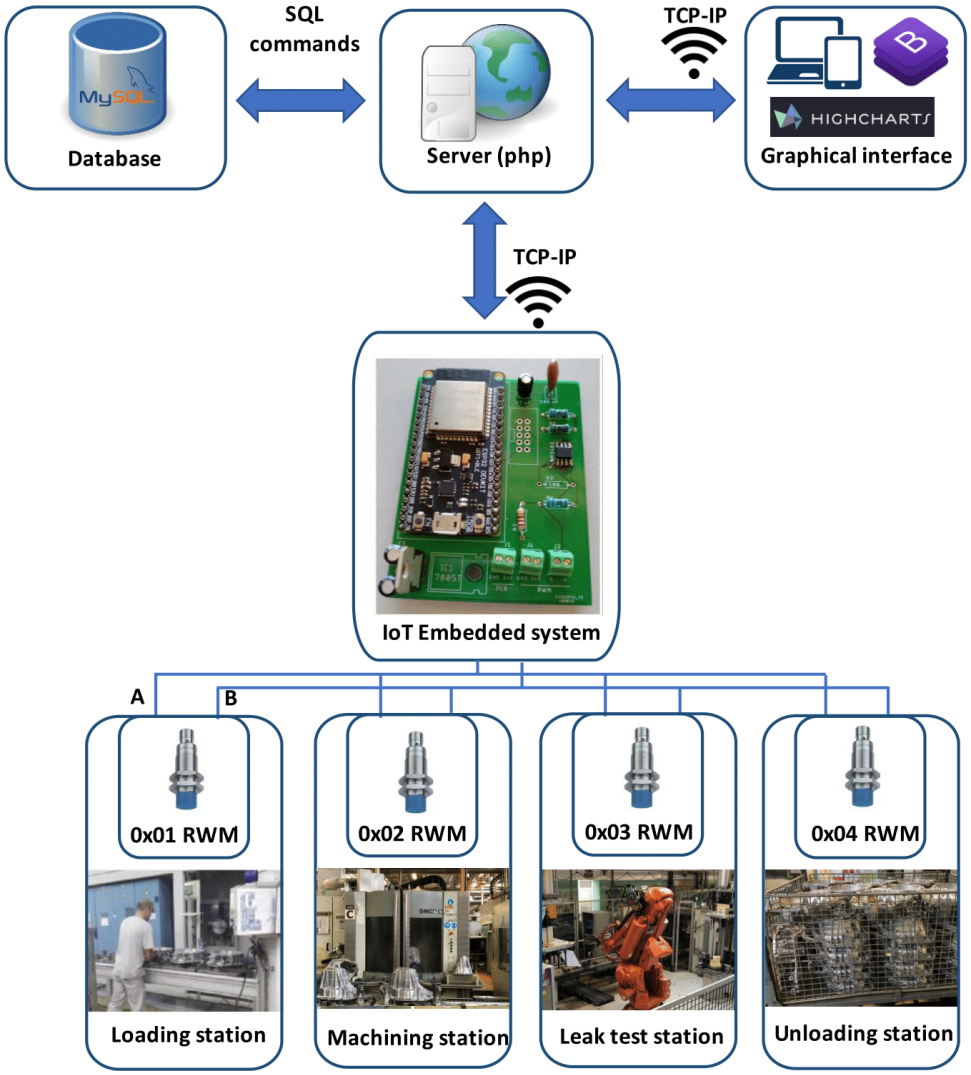


Figure 5.1: Implementation diagram

The implementation of the proposed solution consists of four Contrinex RWMs, each planned to be located at one of four stations in the housing production line.

The first is in the Loading Station, where the raw housings arrive by container. A visual inspection is performed by an operator at this station. If the housing doesn't contain any raw nonconformities, the operator simply loads the housing into the line's conveyor. However, if nonconformities are present, the operator must also use the graphical interface, selecting from the list of expected nonconformities. Upon registering any nonconformities found, the housing is removed from the line. With this implementation, an immediate digital registry is performed, eliminating the need for paper, or atleast adding another, more secure, way of storing data about the individual part.

For this implementation, a reference is assumed by the ESP32 whenever a new tag arrives at the first Contrinex reader. In the real system the display monitor could be used to learn this information.

Another assumption is that a way to uniquely identify nonconform parts after they

leave this first station to QA is implemented at the shop floor. Several choices exist for this to be possible. The operator can continue writing traceability paper records, now containing the serial number and/or the test ID. Another possibility is carrying the housing and pallet to the printer at the unloading station, which would allow to embed a barcode tag to the housing. The operator can also, similarly to what already occurs at the machining centers, write some characters in the housing, such as the nonconformities test associated with the part.

The second station in the line is the Machining Station. The implementation used in this thesis considered only one machining center, partly due to scarcity of available RFID modules, but also due to the prototype nature of this solution. In an actual implementation of this system one ESP32 could manage the messages from several other RWMs positioned on multiple machining centers. However, mainly because processing of messages from these RWMs would be the same, only one was considered as representative of the remaining.

In the Leak Tester Station this implementation considers that the result of the test, processed by a PLC, is communicated to the ESP32. For the present implementation the result of the test was merely assumed, there being no physical connecting between the ESP32 and other devices other than RFID readers in this implementation. The result of the test is both stored in the tag and the database.

The final RWM is situated in the Unloading Station. Here the housings are visually inspected once more, this time in search for nonconformities resulting from the machining. The results of this inspection are stored into the database in the same manner as those on the start of the line, with the operator having to select, on a web page, the nonconformities he found. Housings that reach the final station present no problems in being identified at QA, since the visual inspection is the last operation performed on the housing. Before that the Zebra printer was already used to attach a barcode to the housing. This implementation proposes the alteration of the barcode printed by the Zebra printers. More details are given on this subject in section 5.8.

5.2 Hardware

5.2.1 Contrinex Read/Write modules

The proposed implementation uses Contrinex Read/Write modules (RWM), specifically the RLS-1181-020 model. Contrinex manufactures other readers for industrial use, as previously detailed in section 3.10.3. This particular model combines a small size (73.5x⌀18) and being able to communicate through RS485. Furthermore its communication protocol is ISO 15693, a common protocol for the 13.56 MHz frequency. Another preponderant factor is its relative low price, compared to similar models such as the RLS-1303-020. The latter has a more powerful antenna, which allows it to have a maximum communication range of 60mm, nearly double the 31 mm of the readers used for this solution.

Contrinex readers communicate through the ContriNET protocol, whose message is explained in section 5.2.1.2. The company freely offers a software specifically designed to handle the communication with readers of its brand. This software visually displays the messages sent and received to the readers, as well as the data read from any TAG present in a reader's field. However, this software was not used extensively, since the solution proposed makes an ESP32 take the role of the middleware, processing the messages sent

from the RWM. The company's software was merely used during the initial stages of development of this thesis solution, aiding in quickly exchanging information with the RWM and identifying the commands needed and its structure.

This implementation utilizes these readers due to their low price when compared to the Balogh readers utilized at Renault. Furthermore, Balogh tags also have very high costs when compared to the tags bought for this implementation. Since the hardware costs were covered by Universidade de Aveiro, price was a big factor in the choice of hardware. Another preponderant factor for choosing these readers is the fact Contrinex publicly shares the structure of the ContriNET messages exchanged by the RWMs. Balogh, in turn, only publishes the structure of messages exchanged between the control interface and the controller.

Figure 5.2: Contrinex RLS-1181-020 Read/Write module

The reader's main characteristics are the following [53]:

- Has a threaded stainless steel housing. The reader can be installed by using two nuts, seen in figure 5.5.
- Works with voltages ranging from 14-32 VDC, and has maximum current consumption of 60 mA.
- HF communication, namely 13.56 MHz.
- Compatible with the ISO 15693 standard, being able to communicate with any TAG using this protocol.
- Uses the RS485 standard, allowing for a maximum of 32 devices on each bus.
- Accepts baud rates of 115200/38400/19200.
- Allows to define the reader's address through two methods: Physically or logically, which is further explained in section 5.2.1.1.

5.2.1.1 RWM's addressing

The RWM is identified by its address, which is a single byte. Physical addresses are limited from 0 to 9. They are defined by adjusting the potentiometer built into the reader. Logical addresses are defined by sending a command to the reader specifying the intended address. This method allows for addresses ranging from 0 to 255. The reader has a yellow LED that indicates which mode of addressing is being used:

Figure 5.3: Contrinex RWM's potentiometer

.

- Yellow LED constantly on: the physical address is the one displayed on the potentiometer.

- Yellow LED is blinking: a logical address is being used. Unlike the physical address, a logical address is not visually represented in the RWM. A reader's logical address can be discovered by sending a special command to it without addressing it specifically to the reader.

- LED is off: no address is being used, needing to be defined by either method.

This implementation used 4 Contrinex RWM. All have been physically addressed. Each is located in a significant station of the production line, as can be observed in table 5.1

Address	**Station**
0x01	Loading
0x02	Machining
0x03	Leak testing
0x04	Unloading

Table 5.1: Addresses of RWM's and their station

5.2.1.2 Message structure

The RWM only recognizes and produces messages with a structure compliant with the ContriNET protocol. This structure is shown on table 5.2.

SOF	**SRC**	**DST**	**LenH**	**LenL**	**SeqID**	**CMD**	**CRC**	**EOF**
0x0F	1 byte	1 byte	1 byte	1 byte	1 byte	N bytes	1 byte	0xF0

Table 5.2: Message structure for Contrinex RWMs, adapted from [53]

The bytes displayed in table 5.2 are the following:

SOF - *Start of frame*: the first byte of any message.
SRC - *Source address*: the master's address.
DST - *Destination address*: the slave's address.
SeqID - *Sequence ID*: the sequence identifier.
CMD - *Command*: a byte or series of bytes that identify the action intended for a RWM. In case the message's source is a RWM, the CMD part of the message is reader's response.

LenH and LenL - *Packet length high/low*: The number of bytes contained in the packet of a message. The packet are the bytes contained in the command (CMD) plus the byte correspondent to the sequence identifier (SeqID).

EOF - *End of frame*: the last byte of any message.

CRC - *Cycle Redundancy Check*: Contrinex RWM's use the CRC-8-CCITT algorithm for data error detection.

5.2.2 Contrinex Tags

Contrinex RTP-0501-020 tags were used for this implementation. These tags can be seen on figure:

Figure 5.4: Contrinex RTP-0501-020 tag

Their main features are the following:

- Has a nylon housing with a central hole for fixing a screw.
- Cylindrical shape, with dimensions 3.2x⌀50.
- Passive tags, with a maximum communication range of 31 mm with the RLS-1183-320 RWM.
- Communicate in the 13.56 MHz frequency.
- Use the ISO/IEC 15693 for its RF communication, including anti-collision algorithm.
- Each tag has a unique identifier of 8 bytes.
- Allow 100,000 write cycles, and unlimited number of read cycles. However, data is only kept for 10 years.
- Their EEPROM has a total capacity of 256 bytes, organized in 64 blocks of 4 bytes.

The tag's memory is organized in pages, blocks and bytes. Each page contains 4 blocks, which themselves contain 4 bytes. However, 24 blocks of its memory are used as configuration data, with the remaining 40 blocks available for the user to store data in. Figure 5.5 shows the organization of the tag's data.

The 24 blocks used for configuration contain data such as the tag's UID, passwords and write and read condition of each page. The tag has two possible passwords: one for reading and another for writing. Each page can have its access conditions defined. By default the protection status of all pages is set to 00h, meaning there are no restrictions to reading and writing. Other protections statuses can make use of the two passwords for protecting either only writing or both writing and reading of certain pages.

Pages	Blocks	Byte 3	Byte 2	Byte 1	Byte 0
9	39				
	38				
	37				
	36				
...	...				
1	7				
	6				
	5				
	4				
0	3				
	2				
	1				
	0				
-1	-1				
	-2				
	-3				
	-4				
...	...				
-6	-21				
	-22				
	-23				
	-24				

Figure 5.5: Data structure of Contrinex RFID tags

5.2.3 ESP32

The ESP32 is a Wi-Fi and bluetooth hybrid chip, especially designed for mobile, wearable electronics and IoT applications. It contains two rows of pins, separated by 10 inches, with each pin distancing 2.54 cm (1 inch) from the nearest pins in the same row. ESP32 is a low power chip, needing only 3.3V or 5V of power. Furthermore, the chip has multiple power modes, intended to reduce the chip's energy consumption. These different power modes can disable Wi-Fi and Bluetooth completely, or activate them during certain wake-up events. This chip has Xtensa® Dual-Core 32-bit LX6 microprocessors. By default, only one of cores (Core 0) is used to run the application code, including Wi-Fi communication. The second core must be given specific tasks in order to take advantage of it.

It has embedded a micro USB port that can be connected to a PC to program it. The ESP32 can be powered through this USB connection, making the connection have a double purpose, especially during the testing of simple applications. The chip's official software development framework is called ESP-IDF. Development implementations for this chip, such as Arduino, are based on this framework.

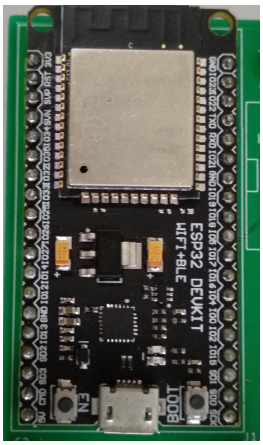


Figure 5.6: ESP32 DevKit V4

The ESP32 has several inbuilt functionalities and characteristics, such as [54]:

- 448 KBytes ROM for booting and core functions
- 520 KBytes on-chip SRAM for storing the application code.
- Functions in temperatures ranging from -40 °C to +125 °C.
- Implements Wi-Fi TCP/IP, full 802.11 b/g/n/e/i WLAN MAC protocol, having an inbuilt Wi-Fi antenna. Therefore the ESP32 can communicate with most commercial routers as a client. The chip can also be used as an access point using this same protocol.
- Integrates a Bluetooth link controller and Bluetooth baseband. Supports both Bluetooth Low Energy and classic Bluetooth, allowing it communicate with a wide range of other devices.
- 2x I2C interfaces
- 2x I2S interfaces
- 4x SPI (Serial Peripheral Interface) in slave and master modes in 1-line full-duplex.
- 3 UART (Universal Asynchronous Receiver Transmitter). These are used to communicate by RS232. The pins for these UART are configurable in the running application code.
- 18 ADC (Analogic to Digital Converter) pins.
- 2 DAC (Digital to Analogic Converter) 8-bit pins.

The ESP32 DevKit V4 was the particular model used for this implementation. It has a total of 38 pins, 30 of which are programmable GPIO's. Some functionalities are bound to certain pins, such as the ADC and DAC, whose pins cannot be reprogrammed. Others, such as UART 1, 2 and 3 can have their pins assigned at will.

Figure 5.7 illustrates the entirety of ESP32's capacities in function block diagram:

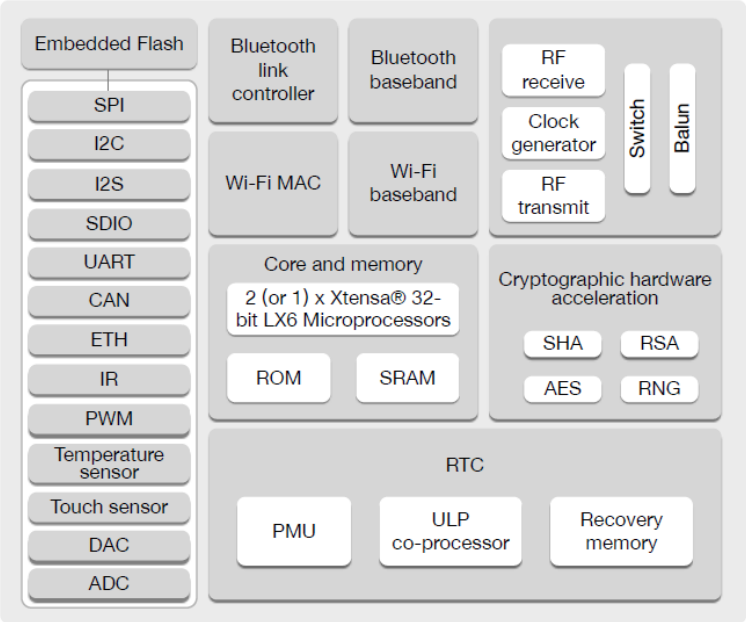


Figure 5.7: ESP32 function block diagram [54]

While the majority of these features were not used, being merely used the ESP32's UART and Wi-Fi, this chip provides a great array of solutions for any system, and opportunities for its expansion. Furthermore, it provides all this for a small price. The model used in this implementation can be bought for less than 10 euros.

5.3 Assembly

Figure 5.8 shows the assembly of the proposed traceability system. The photo shows the four readers utilized, as well as one RFID tag. It also shows the developed embedded system inside a plastic housing. A USB connection is being made in the figure, which is used to program the ESP32. After the application is uploaded to the microchip the USB connection can be ceased.

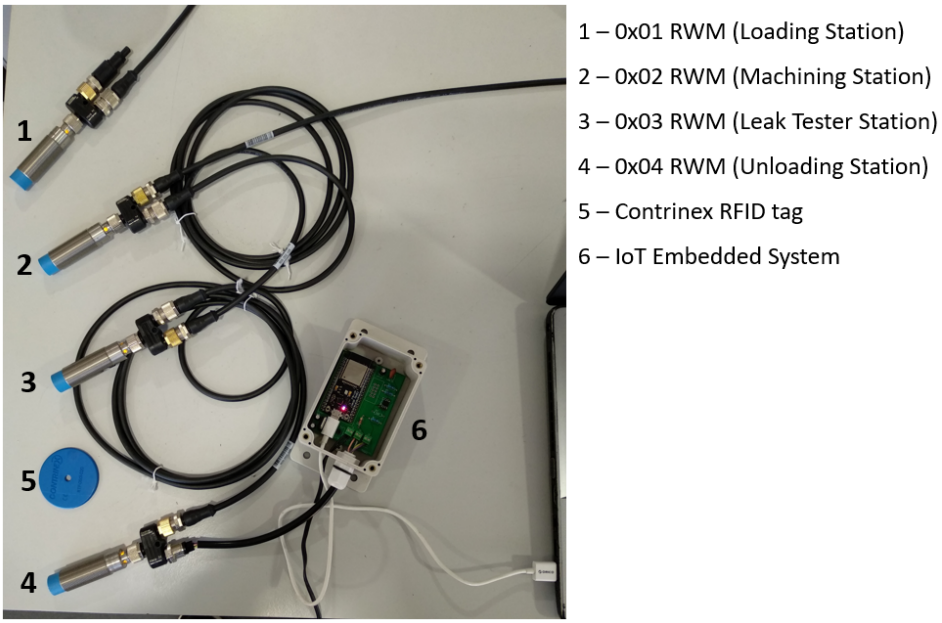


Figure 5.8: Implementation assembly

The developed PCB, both with and without its components soldered to it, can be seen on figure 5.9.

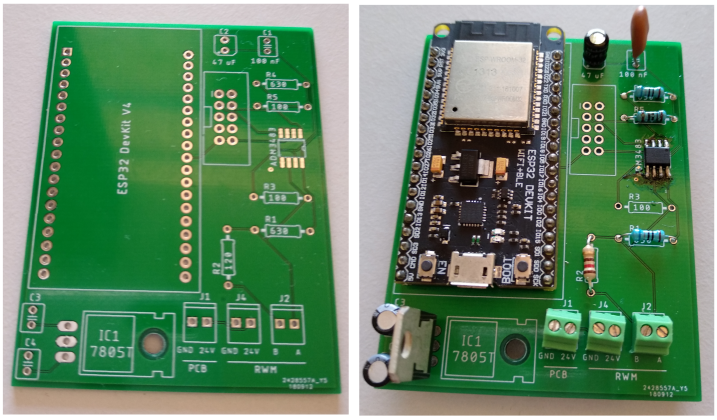


Figure 5.9: Developed IoT embedded system

The PCB was designed in EAGLE. The majority of the parts were already available in the program's libraries, with the exception of ESP32 DevKit V4, whose library was

created for this application. This board uses three connectors, each with 2 terminals. Connector J2 has the A and B terminals of the RS485 communication, to which the Contrinex RWMs connect. To its left is a connector that powers the readers with 24V, which originate from the left most connector on the board, J1.

This connector's supply goes through a voltage regulator, which reduces the voltage to 5V, appropriate to power the ESP32. Two electrolytic decoupling capacitors are also used to stabilize the 5V output of the regulator. The ESP32 then connects with an UEXT connector. The UEXT can be soldered to the board or not. The current implementation considers that no component is needed for it to work. However, the space allotted on the board for this connector allows for further expansion of the proposed solution, allowing another ESP32 or any other device to connect to this PCB.

UEXT itself connects to the a transceiver that converts RS232 into RS485, with the help of two other decoupling capacitors, a terminating and a pull-down resistor, all soldered into the board. The board also uses two 630 Ohm pull-up and pull-down resistors.

Component Code	**Name**	**Description**
J1	24V Input Connector	Receives 24V from outside, powers PCB and J4
J4	24V Output Connector	Powers the RWMs
J2	RS485 connector	Communicates through RS485 with Contrinex RWMs
IC1	7805T Voltage Regulator	Receives 24V, outputs 5V to power ESP32
C3	47 uF Eletrolytic Capacitor	24V Input Decoupling Capacitor
C4	47 uF Eletrolytic Capacitor	5V Output Decoupling Capacitor
E2	ESP32	Communicates with readers and server
J3	UEXT connector	Allows other devices to communicate with circuit
C1	100 nF Ceramic Capacitor	3.3V Input Decoupling Capacitor
C2	47 uF Eletrolytic Capacitor	3.3V Input Decoupling Capacitor
R1	630 Ohm Resistor	Pull-up resistor, connecting A to 3.3V
R2	120 Ohm Resistor	Termination resistor for RS485
R3	120 Ohm Resistor	Termination resistor for RS485
R4	630 Ohm Resistor	Pull-down resistor, connecting B to ground
R5	100 Ohm Resistor	Pull-down resistor, connecting B to ground
S1-S4	RFID RWMs	Four Contrinex RWMs used in implementation

Table 5.3: Components of developed traceability system

5.4 Structure of data on tags

The data stored in the Contrinex RFID tags is presented in the following table.

Data	Memory location	Value
Reference	0 to 9	Depends on type of housing
Serial number	12 to 23	Unique for every housing
Raw nonconformities	24 to 29	O - nonconformity absent X - present
Atelier	32	2 - AT2
Line	33	3 - Line 3
Machining station	34	1 to 12
Drying station	35	1 to 6
Machined nonconformities	36 to 42	O - nonconformity absent X - present
Leak test result	44	O - passed leak test X - failed

Table 5.4: Data stored in Contrinex tags

When compared with table 2.1, which presents the data currently stored in the Balogh tags, several differences can be noted. Firstly, the date is not stored in the tag. Several dates, associated with the several passages through different stations, are stored instead in the MySQL database. Secondly, since every housing is identified by a reference already, this is the identifier stored in the Contrinex tags, instead of associating a number to the housing family, type and variant.

The data stored in the tags was greatly expanded. Each individual housing is identified by a serial number, assigned when the part enters the line. The nonconformities are also stored in the tag. Each nonconformity can either be present or absent, corresponding to bytes with values 'O' or 'X', respectively. While not represented in the table, the order in which they are stored in the tag is the same as presented in section 2.9: bubbles are the first raw nonconformity, while corrosion is the last.

The tag also stores the machining center and drying station it went through, which, alongside the Atelier and the Production Line, fully identify these locations in the factory. Finally, the result of the leak test is also stored.

The data used to automate the distribution of housings to the machining posts remains present in the tag. The drying station the housing is expected to go to is attributed in the loading station. The housing reference is also present, though written as the 10 digit number string defined by Renault. The data in the tags has a structure similar to the one defined by EPCglobal, particularly the first 24 bytes containing the reference and serial number, which correspond to two of the keys of SGTIN. Missing from the tag's stored data is the company's prefix, due to the fact that tags are expected to be used in a closed loop. Therefore the information about the manufacturer is of no significance to Renault CACIA, since it is itself the manufacturer. It is also important to note that despite EPCglobal being taken as inspiration for the data arrangement, the fact that the housing reference, corresponding to GS1's product code, is 10 digits long, is not in accordance with the standards defined by GS1. Should Renault intend to apply GS1

standards to its tags it should shorten the product reference to 7 digits, which, alongside the 6 digits of the company prefix, would total the 13 digits mandated by the standards organization.

5.5 Processing at ESP32 - middleware

The developed ESP32 application starts by running some configurations. It creates several arrays that will store multiple messages. Several bytes are also created and set to default, such as the control byte, which dictates which command is sent to the RWMs at any given time. Secondly, the UART communication is set up. The pins (RX and TX), as well as the baudrate, are defined at this stage. UART1 was used, with a baudrate of 115200 bps. Another major functionality configured in the setup is the Wi-Fi communication. The ESP32 is connected to a wireless network, using the network's SSID (Service Set Identifier) and its password. It then connects to the IP of the computer that hosts the server. After that the ESP32 is ready to communicate with the pages present in that server. The first request performed aims to find the latest serial number attributed on the production line. The application running in the microchip adds a unit to the last attributed serial number, assuring numbers are not repeated.

The UART and Wi-Fi communications were allotted to two different tasks running on two different cores. The two tasks have their own loops that run simultaneously, while sharing the same variables. This allows for the two communications to be as fast as possible. The choice to separate the two became a necessity, especially because the high number of HTTP requests hampered the responsiveness of the system. If both communications occurred on the same core, the ESP32 would have to wait for the Wi-Fi communication to be performed successfully in order to send commands to the RFID modules. The developed application takes around 0.8 seconds to perform a HTTP request, which initially made the detection of RFID tags very unreliable. Assigning the UART communication, as well as most of the processsing, to Core 1 allowed communication with the RFID modules to become incredibly reliable, regardless of how many HTTP requests are being performed simultaneously.

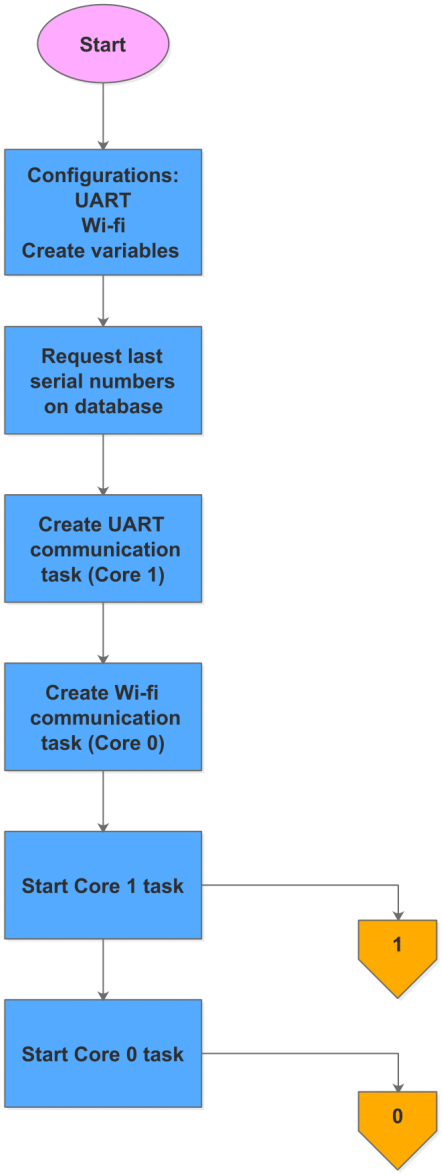


Figure 5.10: ESP32 processing - Configurations and task setup

It is important to understand the general architecture of the application before going into greater detail. Particularly, the communication between the ESP32 and the RFID RWMs, which obeys a general rule. The running program's objective is to send Status commands as often as possible to the four readers in the bus, in order to quickly respond to the presence of a tag in a reader's field. To that end it cycles through the four readers whenever no tags are present in either. However, once a tag is found in any of the readers, that reader and tag become the priority of the communication. A 'command cycle', defined as all the commands needed to both read and write data, is performed, and only afterwards does the communication with the remaining readers resume. Figure 5.11 showcases the communication with the readers in a simplified manner.

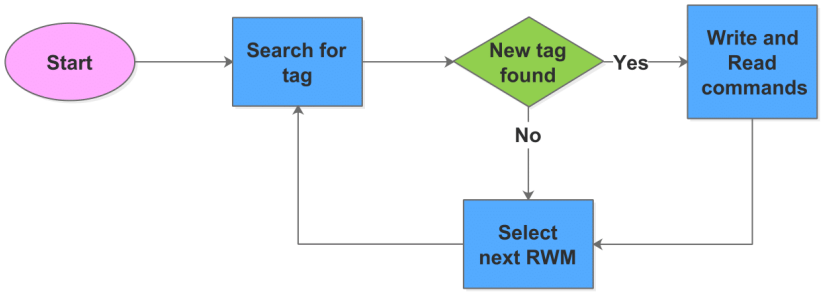


Figure 5.11: ESP32 processing - selection of next RWM in the line

Taking this into account, the application can now be described more fully. A command is sent every 70 milliseconds, which necessitates GPIO5 to be set to 1. After the order is given to send data through the UART1, a delay must be written into the code, since the ESP32 UART doesn't stop the remaining code from running when it is sending messages. This delay must be long enough to send all the bytes in the message.

After the message is sent, GPIO5 is set to 0, and the processing unit will be ready to read the incoming responses. ESP32 processes data received in its UART byte by byte, with the arrival of each byte causing an interruption. Whenever a byte is detected, an algorithm starts to run. The first thing it tests for is if the byte received has a value of 0x0F, indicating the start of a message from the Contrinex modules. In that case, this byte is stored in a temporary array, separate from another array that will contain what is expected to be a full message.

The ESP32 then expects a second byte with the address of the RWM. Once this byte is received, it stores these first two bytes in the definitive array. The remaining of the message is received, up until the final byte of the message, 0xF0, arrives. The developed code predicts the eventuality of communication errors. Because even if a byte with value 0xF0 is interpreted by the UART, it doesn't necessarily mean a valid message was received. The first test performed on the message consists of comparing the expected length of the message, which corresponds to its fourth and fifth bytes, and its actual length. In case the two match, a second test is performed. The error detecting code CRC-8-CCITT is used on the received message. If the calculated byte corresponds to the CRC in the received message, then it is valid.

Its important to note that the developed application has some limitations regarding the reading of bytes in the UART. Initially the application tried to read all bytes in the UART in the same loop. Unfortunately this proved unreliable, with several bytes being wrongly interpreted by the ESP32. Therefore a new byte is read every loop, which takes 1 millisecond. The 70 milliseconds it takes to send a new command to the bus include the time it needed to read the longest response (the Read command) plus several other delays that occur in the processing of the response.

The two following flowcharts (figures 5.12 and 5.13) show the major processes and decisions coded into the ESP32 for writing and reading messages.

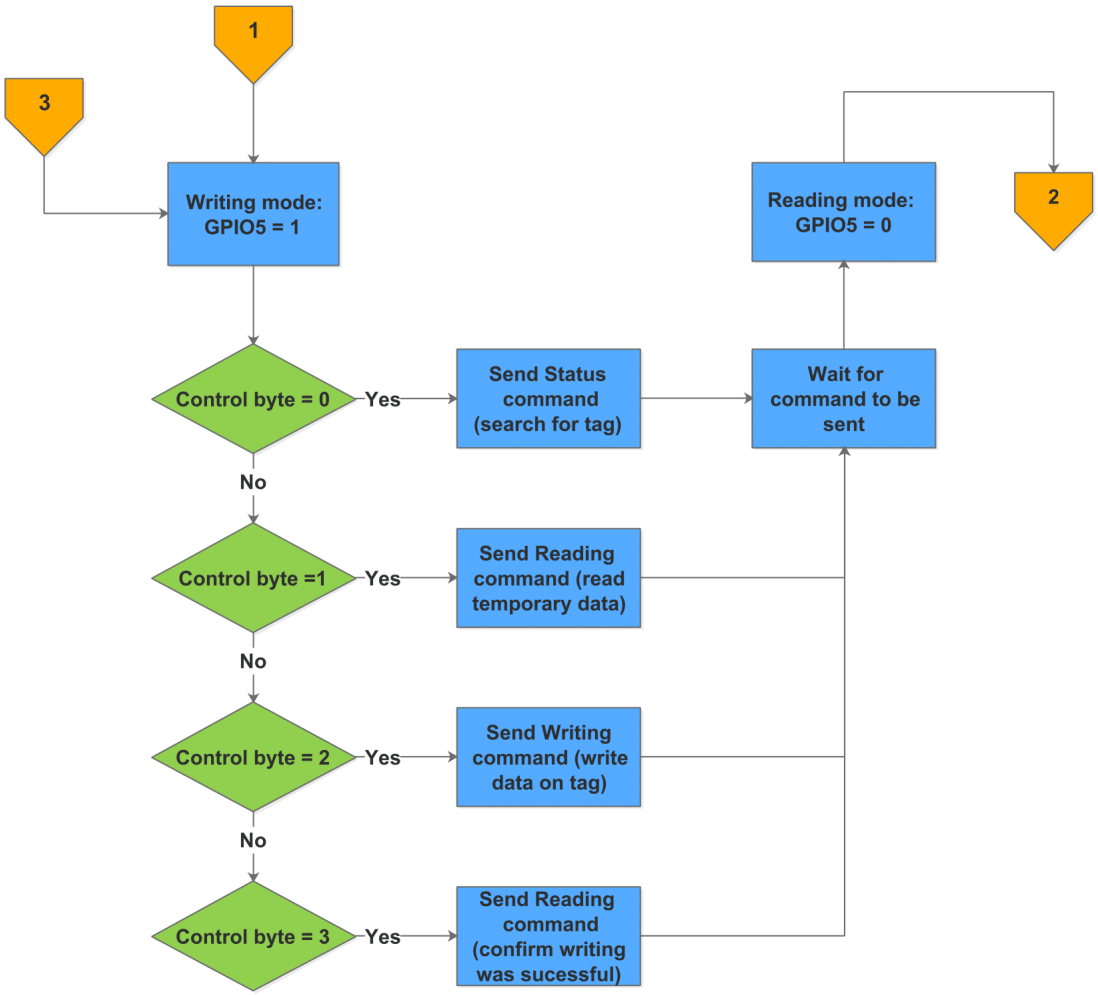


Figure 5.12: ESP32 processing - commands sent to RWM

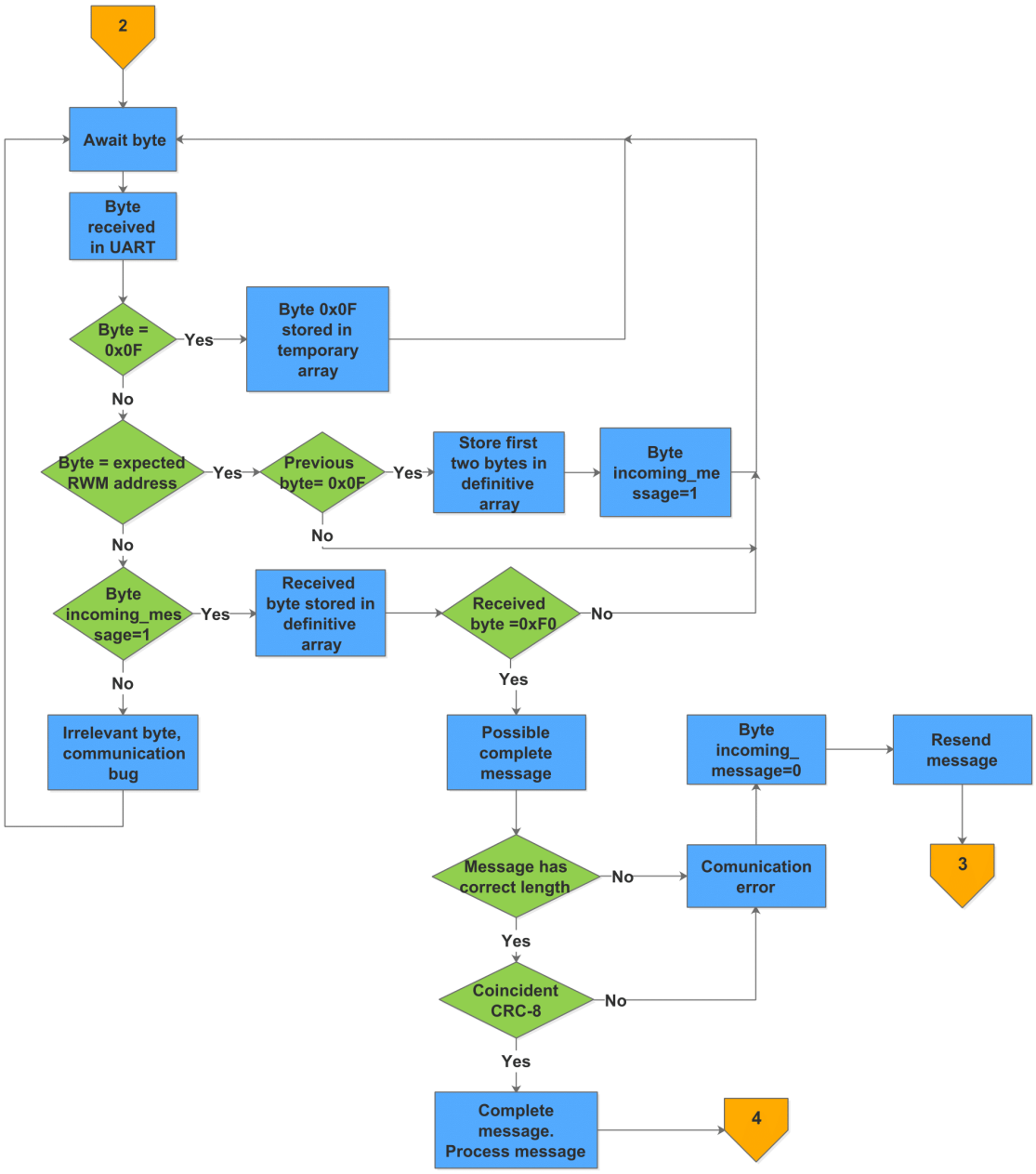


Figure 5.13: ESP32 processing - reading responses

Once a valid message is received at the UART, it must be processed. If the message received corresponds to the answer for the Status command, it can either indicate that a tag is present or not. When a tag is present the message will contain the tag's UID, which must be stored in a variable to be used for the subsequent commands.

In the scenario a tag is found for the first time in the RWM's field, the code then orders the ESP32 to send a Read command, changing the control byte to 1. This time the answer will contain several more bytes of information, including several blocks of the tag's memory. This first read command is used to store temporary data that can be changed in the following write command. But most of all, it is included in case only a

part of a block or blocks needs to be changed. Since an entire block needs to be written again even if only some of its bytes need to be changed, reading the data that needs to be preserved is absolutely necessary.

The following command is the writing command. No major processing is used on the answer to this command, as it simply indicates whether the writing was successful, which corresponds to the Acknowledge byte being 0x00. Regardless, a second confirmation is performed with the last command in the cycle, a read command. If that is the case, the cycle is considered complete. Otherwise, the ESP32 will order the RWM to write the intended data into the tag once again. Once the cycle is completed, the ESP32 sends data to the server. This data will be shown in the graphical interface.

After the four commands of a cycle are completed the tag will very likely remain in the field. In that case, a new byte is activated, making the RWM continually search for a tag. Meanwhile, the ESP32 shifts its attention to the remaining RWMs, which have not been queried for the 350 milliseconds it takes to complete all five commands on a reader (Status+Read+Write+Read+Status). Only once the tag leaves the field, is the command cycle restarted.

While the previous operations occur on Core 1, Core 0 has been performing any necessary HTTP requests to the server. One such request occurs whenever a command cycle is completed. In the presented implementation, only the RWMs in the loading and unloading stations send this type of data to the server, as they are the only stations that have a dedicated graphical interface needing real time data.

A separate request happens approximately every 500 milliseconds, and is looking for changes in the values of the nonconformities pertaining to these same stations. Initially the writing command assumes the part has no defects. Only if the operator finds any, and uses the graphical interface to indicate those nonconformities, does the ESP32 use those new values to rewrite the values it previously assumed. This entails forcing the control byte of the appropriate RWM to 2, making the ESP32 send the write command once again, now with new information.

The third type of request occurs after the tag leaves the module's field. The fields on this request vary depending on the RWM. They are sent to a page, RFID_contrinex.php, which handles the entirety of the database management. It is important to notice that every single one of these requests can fail, in case the Wi-Fi connection is unsuccessful. In that case, the ESP32 is ordered to restore the connection, and then perform the request once more.

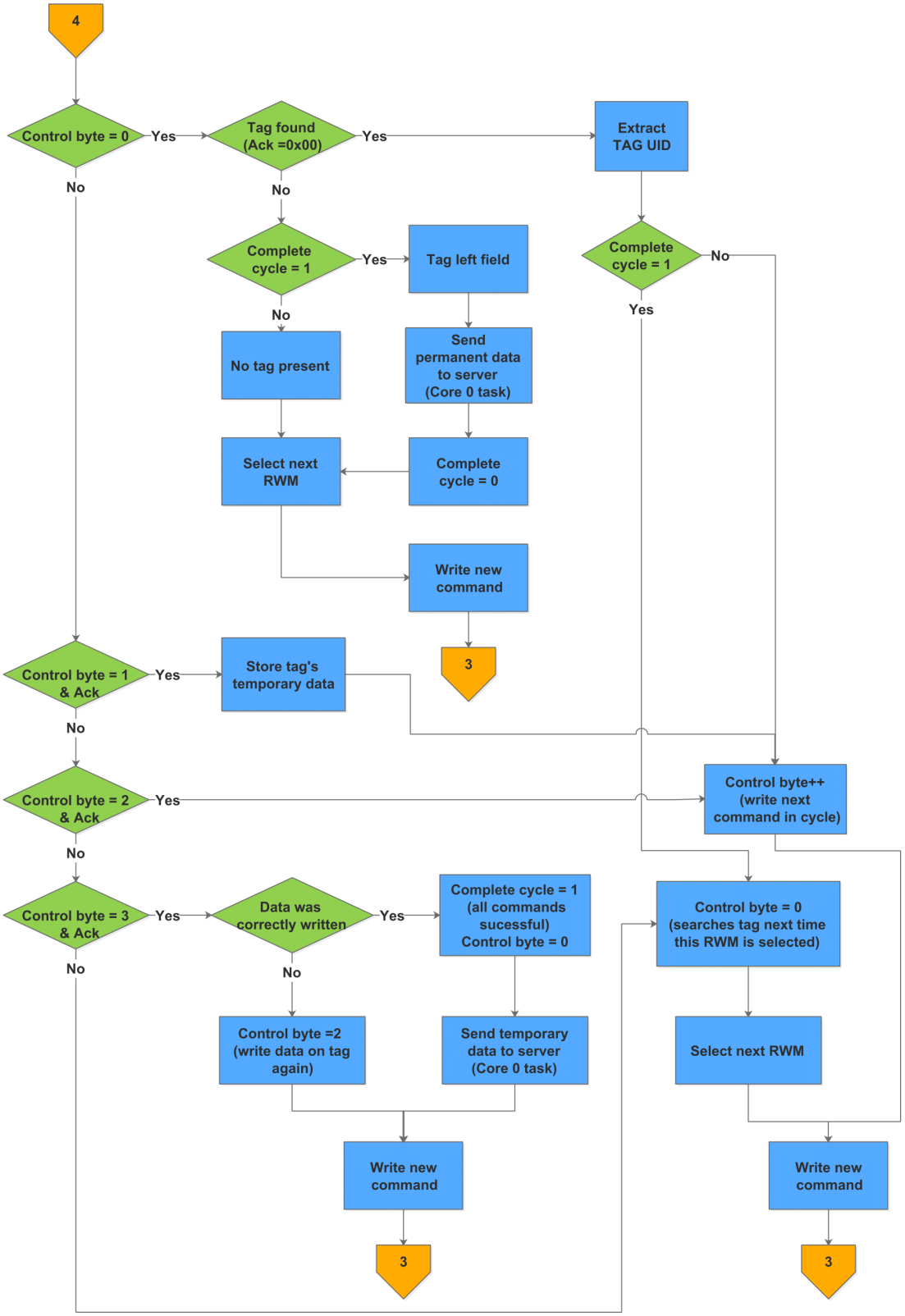


Figure 5.14: ESP32 processing - processing responses

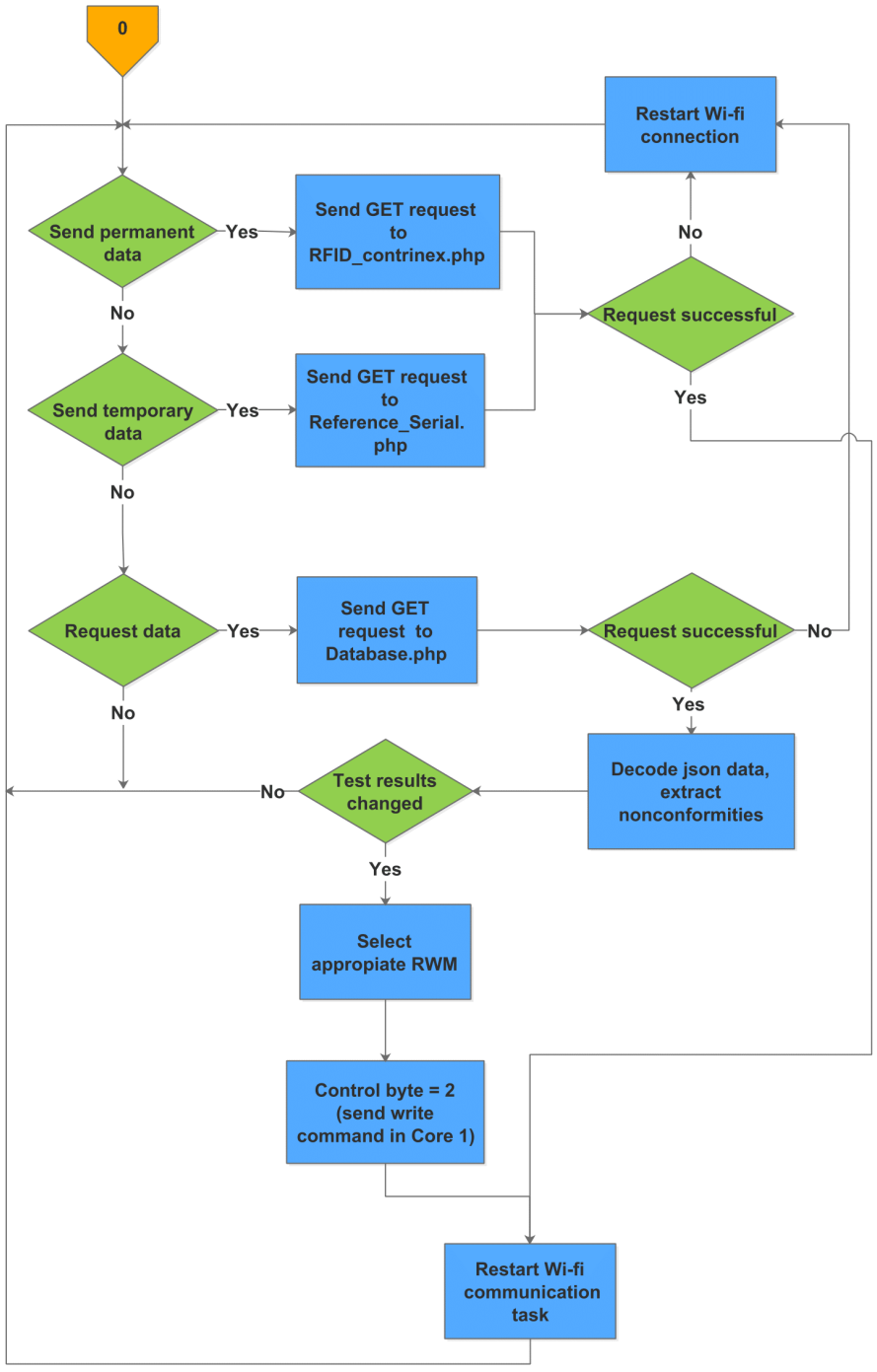


Figure 5.15: ESP32 processing - HTTP requests sent to server

5.6 Processing at Server

The server side of this system contains several files. The most important among them is the RFID_contrinex.php file. This page is expecting requests containing information that is meant to be stored permanently in the database. The page is expecting four different messages from for four different RWMs, and will act differently according to the source of the information.

The first RWM corresponds to the highest number of Queries performed to the MySQL database. The first query searches the containers that are currently being loaded into the line, as well as the reference of each one. If the reference it received from the ESP32 request matches any of the references currently being loaded, this page associates the serial number with the matching container's ID. The page then inserts the housing's serial number into the Housings table, as well as its reference and arriving container ID. Next, data needs to be inserted in both the Loading Station table and the Raw Nonconformities Table. When these queries are made, the database creates an ID for the entries on both these tables. These ID's are the primary keys of these tables, and are obtained, by the server, by performing two other queries. Due to the fact that the Housings table contains these keys as foreign keys, this table is updated with these unique ID's. Lastly, since this HTTP request from the ESP32 coincides with the departure of the tag from the Loading station, some final queries are performed. The update tables, which serve merely to store temporary data used to update web pages in real time, have their values of the reference, serial number and raw nonconformities, set to default.

A HTTP request from the RWM corresponding to the Unloading Station leads to similar queries, with two major differences. Data is not inserted into the Housing table, only updated. And the Nonconformities Test ID, in this case pertaining to machined nonconformities, is updated into two different tables, the Machining and Loading Stations, since this test's ID is relevant to both stations.

The other two RWMs lead to far fewer queries, inserting only data into their respective station's table, and updating those entries' IDs into the Housing table. In the tables of all station passages, one of the fields concerns the date of said passage. This is the only field that is not extracted from the GET request, instead being obtained by the php file. The date is obtained according to the coordinated universal time (UTC) for Portugal.

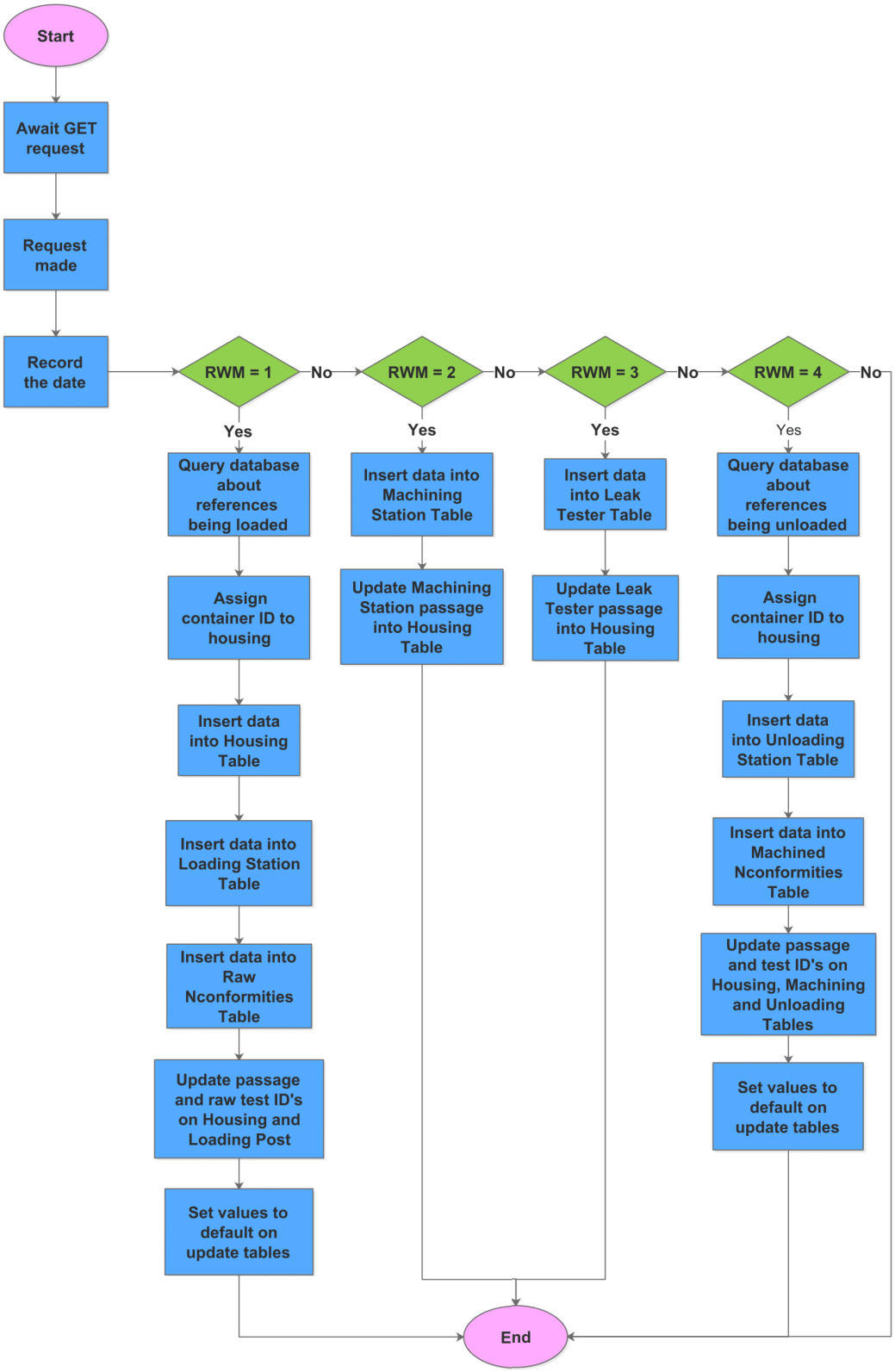


Figure 5.16: Server processing - Data received from ESP32, processed and sent to database

The remaining pages present in the server are the following:

- Reference_serial_number.php : It receives GET requests with the temporary data needed to display in the graphical interface. The data it receives from a station is the serial number, the reference and the machining center. Two stations can make these requests: Loading and Unloading. The data is then shown in the pages Raw Products Nonconformities and Machined Products Nonconformities (explained further in section 5.7.

- RFID_line.php: Creates a set of data with the production over time. It does so by querying the Housing Table and counting the number of parts of a certain reference every time a new part is produced. The date used for each set of data is the one from the passage at the Unloading Station. Furthermore, housings are only considered manufactured if they passed through all stations and if no nonconformities were found. This production data is used in a linear graph in the graphical interface. This page also creates a different set of data to be used for creating a bar graph showing the total production of each reference.

- First_station.php and Last_station.php: Count the total number of each nonconformity, to be displayed in two separate graphs for each of the tests: First_station.php for raw nonconformities, and Last_station.php for machined nonconformities. Another function these pages perform is to compile the list of housings of the containers that are currently being emptied or loaded, giving this information to the Arriving and Departing Containers pages.

- Database.php : This is the page the ESP32 sends an HTTP request every 500 milliseconds when tags are present at either the Loading or Unloading Stations. Queries two update tables for raw and machined nonconformities. The values in these tables are the ones set by the operator in the graphical interface after performing a visual inspection. This process and its graphical interface is further explained in the next section. The page arranges the data into JSON format which is then interpreted by the ESP32.

- Last_serial_number.php : Queries the Housings table and returns the last assigned serial number. This page is used by ESP32 during its setup.

5.7 Graphical Interface

The graphical interface is a set of web pages developed using a open-source framework called Bootstrap. This framework includes HTML and CSS design templates for several graphical elements such as buttons, tables and navigation bars. It also implements a grid system, composed of rows and columns. Each row contains twelve columns, and each cell of this grid can contain its own separate interface elements, separated visually from the remaining cells. This grid will adapt to the size of screen, making Bootstrap an excellent tool to design web pages with smaller devices, such as smartphones, in mind.

For creating charts, Highcharts was chosen, which uses JavaScript to create interactive and dynamic charts. These can be updated by JavaScript functions, showing information

in real time. Bootstrap doesn't offer its own tool for this purpose, but is fully compatible with Highcharts (other JS libraries could have been used instead, such as ChartsJS or CanvasJS).

The graphical interface contains several web pages. Its main page can be seen in the following figure:

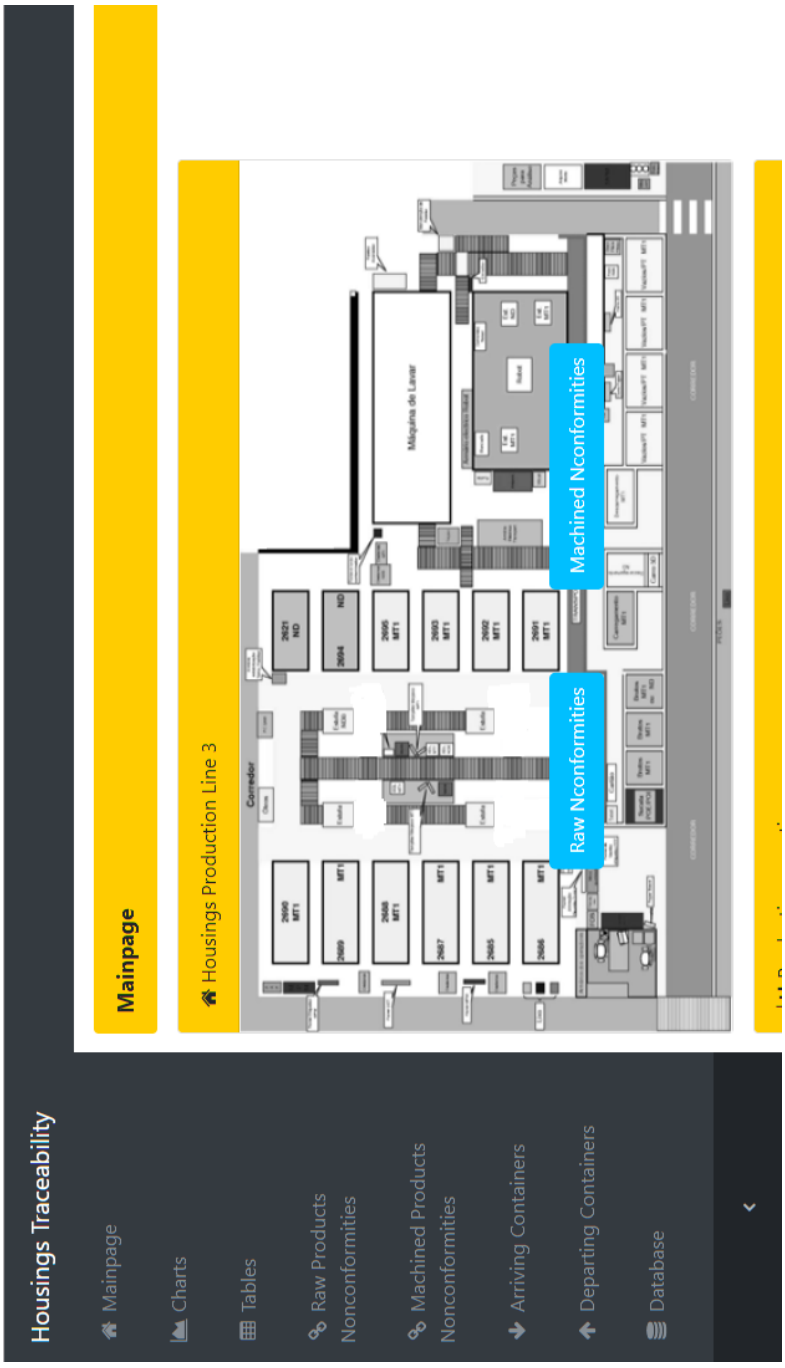


Figure 5.17: Graphical interface - Mainpage

On the page's left is a navigation menu, containing hyperlinks for the remaining pages

of the interface. As can be seen, the Mainpage itself has a representation of the production line, with one button that sends the user to the Raw Products Nonconformities page, and another, on the left, to the Machined Products Nonconformities page. Represented in figure 5.17 is only the top of the page, corresponding to one row of the page's grid. The remaining rows contain several charts, allowing the user to visually see data about the production in this main page.

Another page, entitled Charts, contains only the charts, serving as another viable option to visualize the total production and number of nonconformities detected in the production line.

The Raw Nonconformities Page contains a table in which the operator can select the defects it finds in a housing.

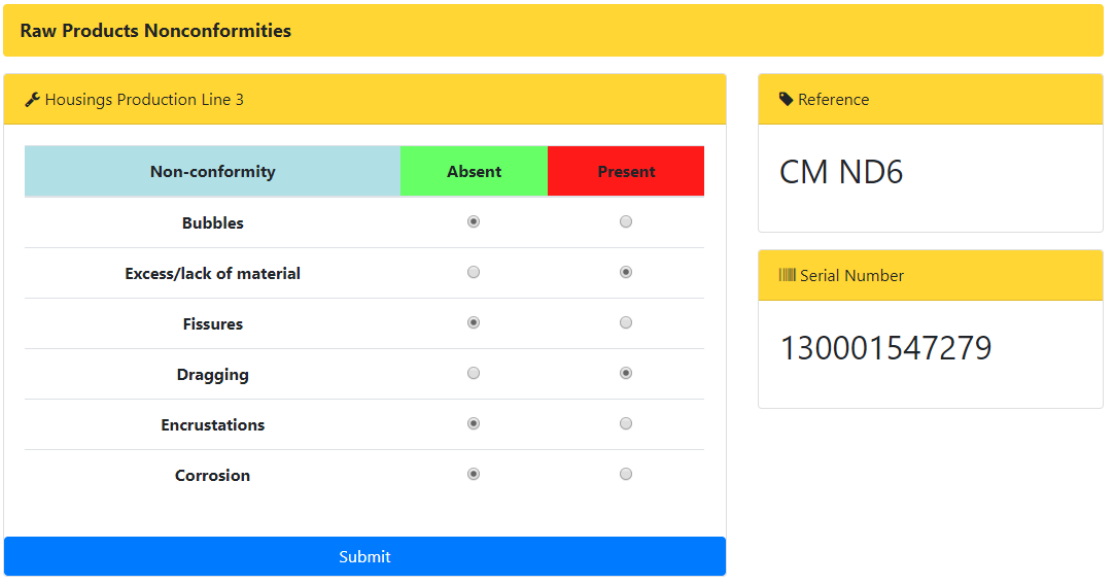


Figure 5.18: Graphical interface - Raw Products Nonconformities

In the right portion of the page two boxes can be seen, each showing the reference and serial number of the part present in the Loading Station. In case no part is present, these assume the value None. When this is the case, the operator cannot select any of the 'present' radio buttons on the nonconformities panel. Only when a tag is present in the RWM's field can the operator select the nonconformities. He should then click submit, which will force the page to perform a query to the database. These values are updated in the update_raw_nconformities table, which is then accessed by the Database.php file. This php page relays the information back to the ESP32. As previously explained, the ESP32 will test if there was any change in the nonconformities since its last request. If so it writes the nonconformities test results into the RFID tag.

After the operator clicks submit the nonconformities panel will retain the values the operator selected. Only when the tag leaves the field are the buttons set back automatically to 'absent'.

Further down the page are presented charts detailing the number of nonconformities found in the production line so far. These graphs are updated in real time. To achieve this the web page's file contains JavaScript code in which an HTTP request is performed

with the AJAX method. The GET request is made to the First_station.php page, which will access the database and compile the data in JSON format. The data is then decoded by the JavaScript code, and the chart is refreshed whenever any of its data changes. This same request is used to obtain the real time values of the reference and serial number.

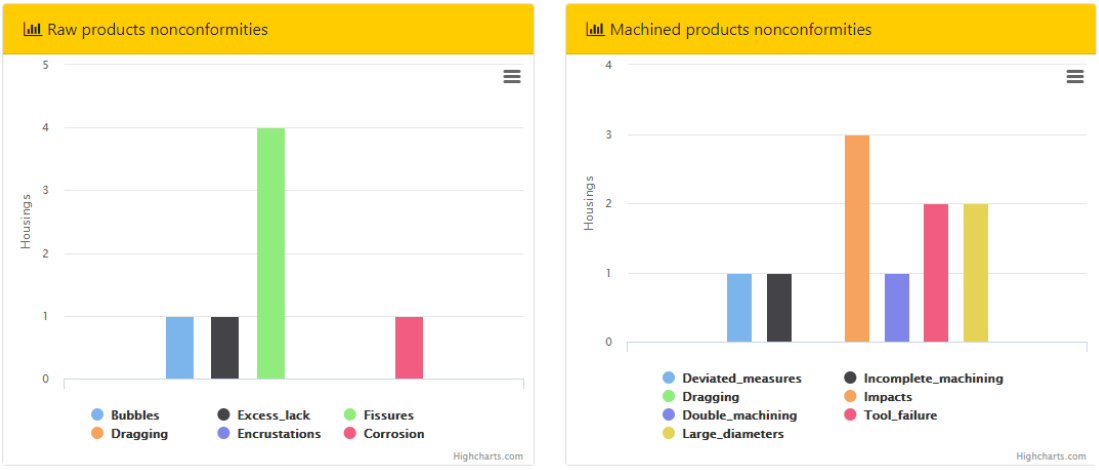


Figure 5.19: Graphical interface - Nonconformities charts

The Machined Nonconformities page is very similar to its Raw Nconformities counterpart, with its own nonconformities on a submit panel. It shows the reference and serial number of the part in the Unloading Station, also showing the machining center it went through during the production. The charts it shows are the same as the other page as well. This philosophy used was the operator or manager shouldn't have to change page to so data relating to the page they're currently on. So, for that reason, the total production and total number of both raw and nonconformities are available at Mainpage, Charts and Nonconformities pages. The vertical navigation bar is present in all these pages, making it easy to move between pages in any order. Also of note, this navigation bar can be minimized, with each page hyperlink being an icon.

The Arriving and Departing Containers pages were created to be used by the operators at the Loading and Unloading Stations as well. Figure 5.20 shows the interface for the Arriving Containers page. These pages were developed due to the need of associating each gear housing to the containers they arrive and depart on. In the case of the Arriving Containers page, the operator should insert on the appropriate fields the container ID (its GALIA number), the housing's reference and the quantity the container carries, information which is available on the GALIA tag.

The page has entries for only two containers, as usually no more than two containers are being loaded into the line. Furthermore, no two containers of the same reference are loaded at the same time. When the operator loads a housing of a specific reference onto the line, that reference is communicated to the ESP32, which in turn transmits to the server. Since the web pages have the information about the containers currently being loaded into the line and their respective references, they can assign the housing to the arriving container it belongs.

A more advanced version of this solution would include the operator merely scanning the GALIA ID barcode instead of manually entering the code in the web page. However,

due to complications with the available barcode scanner, this functionality was not developed. It's important to note that the page sends an alert when a container is full and tells the operator to create a new container when ready. The page also contains a list of all the housings loaded into the line from each container, which is updated in real time.

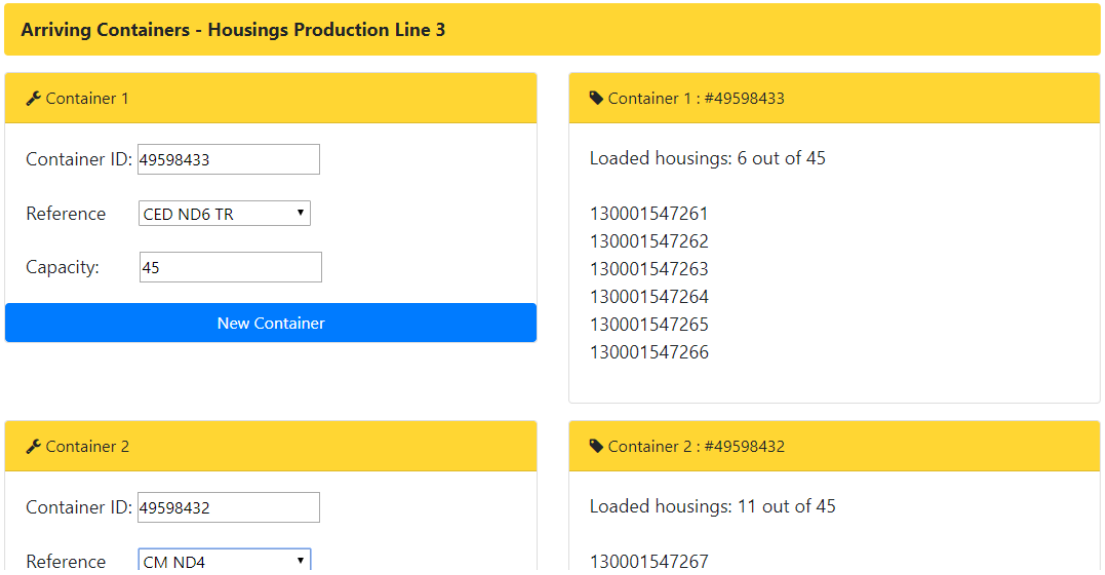


Figure 5.20: Graphical interface - Arriving containers page

The Departing Container page has exactly the same structure. However, here the operator only inserts the reference of the new container it wants to create. Currently the operator creates a new container by, similarly, only entering a reference in PSFP. The Renault system creates a GALIA tag with all the parameters, including the container's capacity. When the container is full, a GALIA tag is printed and its data stored. Similarly, the developed implementation creates a new container with the reference submitted in the web page, assuming the GALIA's remaining attributes. While the developed page allows to associate every housing to a departing container, it lacks the connection to the Zebra printer that PSFP currently implements.

The remaining two pages of this interface concern the raw data contained in the databases. The Tables page can display on the Bootstrap created page the tables contained in the database. The Database page allows for a more detailed access of the database, using the open source software phpMyAdmin. This software creates php pages that show all the tables in a user's database, while also allowing to search for specific values (such as a serial number). Tables can also be altered or created in this page.

More information about the created web pages is available in appendix A.

5.8 Traceability after production line

The developed solution's main intent was to create a closed loop traceability system for Renault CACIA's production line. While not the major focus of this thesis, the traceability of the housings after they leave the production line at Cacia is of the uttermost importance, since the ability to trace back the housings is only possible if any given

housing can be uniquely identified in the subsequent stages of production. Currently, a paper 1D barcode is applied to each housing leaving the line, containing the reference of the housing. Once it arrives at the assembly line, the barcode is read. Since only the reference is present in this code, traceability is limited. The database associates an assembled gearbox to the reference of the housing, as well as the container it arrived in, but this does not allow to trace the part back to the machining center it went through. For that the serial number needs to be present in the paper tag. Figure 5.22 shows some of the considered barcodes, as well as the currently used barcode.

Barcode	Data	Dimensions (with Quiet Zone)	Symbol
Currently used (Code 128)	Reference: CM8200667174	44.2x18.0 mm	
Expanded (Code 128)	Reference + Serial number: CM8200667174.105412546025	68.5x18.0 mm	
Serial only (Code 128)	Serial number: 105412546025	44.2x18.0 mm	
Datamatrix	Company Prefix + Reference + Serial number: 0614248.CM8200667174.105412546025	10.0x10.0 mm	
QR Code	Company Prefix + Reference + Serial number: 0614248.CM8200667174.105412546025	11.4x11.4 mm	

*Quiet zones for barcodes with the same measures as currently used in Renault CACIA. Quiet zones for 2D codes were chosen in accordance with GS1 rules (at least three times the width of a module)

Figure 5.21: Possible barcodes for proposed traceability system

The 1D barcode containing both the reference and serial number presents much bigger dimensions than the one currently used. This solution is not viable for some housings, due to limited space in which to apply the label. Another solution would using the barcode just for the serial number. The housing would be uniquely identified, and since the housing is associated with a reference in Renault CACIA's database, the reference could still be read at the start of the assembly line. However, the alternatives which present the most room for improvement are the ones using 2D codes, namely Datamatrix and QR Code. Both of these tags can still be printed by the Zebra printers present in the production line, requiring only to alter the program running on these printers. These codes can codify both the reference and the serial number, and go further, codifying the company's prefix, as defined by GS1. However, while these codes follow the structure of SGTIN, they are not completely compliant with its standards, since the reference used has 10 digits, which together with a GS1 assigned company prefix would total 16 digits.

This is more than the 13 digits required by GS1. However, should the references have their length redefined, the proposed solution would oblige the GS1 standards.

Paper tags are a cost effective way of tracking parts between facilities, but cannot be expected to remain applied in the housing when the finished product, a car, is sold. Laser marking the code into the housing would solve this problem. A fiber laser marker such as AREX 50, already used at other production lines is a possible solution.

Lastly, the structure of the envisioned serial number has some important specifications. The first two digits of the number identify the production plant and the module. Module 3 of Cacia takes the number 13. The remaining numbers are variable, and are an integer that is constantly added up whenever a new serial number is created. This way, if four different ESP32 are present at the four production lines, there is no need to constantly query the database about the latest assigned serial number, as two housings produced in different lines can never have the same two numbers. It also prevents two housings from different plants having the same serial number.

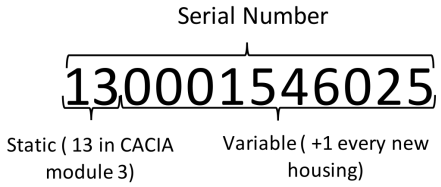


Figure 5.22: Proposed serial number structure

Chapter 6

Conclusions

Traceability is of extreme important for any entity that deals with products, and a company responsible for manufacturing car parts such as Renault CACIA is no exception. The knowledge of the location and history of any given part is of immense value for a company like Renault. One of the main reasons to implement a traceability system is the ability to trace back a defective part's origin, the machining centers it went through, alongside the exact moment the machining has occurred. The advent of Industry 4.0 pushes the concept of traceability forward, with the integration of the IoT in smart factories. To this purpose smart objects, capable of holding information about themselves, are needed. This data, constantly collected on the shop floor, must be shown to the factory's employees through web services, allowing to monitor and influence these smart objects.

Of all the AIDC technologies, RFID presents the most potential. Its ability to store large quantities of data in tags resistant to manufacturing conditions is a major upside. RFID communication also happens without physical contact, and without the need for a line of sight between the reader and the tag. Anti-collision algorithms are already present in most RFID communication standards, allowing to read several tags simultaneously, and with a powerful set of antennas in both reader and tag, communication can occur at considerable distances. The reading of tags is also usually done automatically by stationary readers, allowing the auto identification operation of objects to be fully automated.

This dissertation set out to propose and implement a traceability system capable of handling the needs of the housings production line of Renault CACIA. RFID AIDC is a core part of this solution, with four Contrinex readers being considered, each utilized at a different station of the production line. One of the main objectives of this proposal was the use of a cheap alternative to RFID control interfaces. The utilized alternative was a PCB containing an ESP32. This solution has a cost of less than 20 euros, which is huge reduction from Contrinex or Balogh PROFIBUS interfaces, which would cost 770 and 1490 euros respectively. This embedded system is capable of communicating with several readers at once. Though the current implementation uses only four readers, it has the possibility of connecting several more readers into the same bus.

The communication between the ESP32 and the readers was implemented successfully. An application capable of sending commands to the different readers was developed for the ESP32. The application continuously searches for new objects in the stations it controls, and whenever a new tag is found, all read and write commands are performed.

The achieved communication between the ESP32, the readers and the tags is completely reliable. It manages to perform all commands in 280 milliseconds after it finds a tag in an RWM's field. Since the ESP32 is constantly cycling through the four RWMs, the response time can be as long as 560 ms when it needs to query the three other readers when these happen to have no tags in their fields. However, if the other readers do have tags in their fields, reaction time increases significantly. The worst case scenario has the application taking 1400 ms to read and write a tag in a particular station when all the remaining RWMs detect tags before this station is again queried by the ESP32.

The proposed system greatly reduces the need for physical networks, such as PROFIBUS, to communicate with the company's information systems. Instead the ESP32 communicates through the Internet with a host computer. The computer houses a server, containing several php files programmed to process the information sent from the ESP32. This server arranges data received from the ESP32, such as the housing's serial number and its current station, and communicates it to the major information system developed, the MySQL database.

This database was conceptualized using an ERD, from which the relationships between the different entities were normalized and translated into tables. A total of 11 tables were created. The current implementation inserts and updates 8 tables: Housing Tables, the four Stations Passage Tables, the Departing Container Table and the two Nonconformities Tests Tables. The remaining contain mostly static information, such as the Reference and Supplier Tables, or information that is updated from systems that are not implemented in this dissertation. The created database fulfills the needs for a complete traceability system. It keeps record of individual housings, including the location, date and hour of the passages through all stations, as well as the nonconformities found in any given part.

Web services were also developed. These serve several different purposes. One is to visualize data about the total production in the line in a concise manner. Graphs were created in Javascript to relay data to the operators or managers of the line, using the normalized information contained in the database. These services also allow to monitor some stations in real time, such as the loading and unloading stations. In some web pages real time information about the housing is displayed. These same pages also allow to select nonconformities found in the housing. This information is transferred through the traceability systems's multiple devices all the way to the RFID tags. The fact an operator can save specific data into the tag through web services opens a range of possible automation applications, tough in this dissertation this nonconformity selection operation serves mostly for traceability.

These web services were also developed anticipating the use tablets and smartphones by the line's employee's. A dedicated framework was used to create these pages, which adjust themselves accordingly depending on the size of the device screen. The web services also allow to access the database, protected by a login and password.

The Wi-Fi communication proved to be very reliable, not having been recorded any loss of data in the final version of the application. The communication speed was improved significantly after starting to use both cores of the ESP32. The update rate of real time elements of the graphical interface is reliably no longer than 1.5 seconds, and generally takes 1 second.

Though some architectures already exist for traceability systems, such as EPCglobal, the proposed architecture merely took inspiration from these, applying it specifically to

Renault's housings production line. Therefore, the major objectives of this dissertation were achieved. A cheap and effective hardware was used as a middleware to communicate with several RFID transceivers located in several stations. A normalized database was conceptualized and implemented, alongside web services that use real time and historical information to convey information to the workers in the factory. However, the work developed could be improved in several ways, explained in section 6.1.

6.1 Possible improvements

The implementation presented in this dissertation has several limitations. Firstly, it is a prototype that was never implemented in the real life production line it is based on. It also assumes the ESP32 receives several data, such as the reference and leak test result, without implementing the communication with the devices that have this data. An improved version of this system would include more devices communicating with the ESP32.

Translating this prototype version into a real application would also test the tags and readers performance in metallic environments. Contrinex manufactures RFID readers and tags especially designed to work in metallic environments. However, since the hardware's costs were covered by University of Aveiro, that fact didn't allow to buy more expensive RFID hardware. While the manufacturer assures that this HF hardware can perform in metallic environments as long as a spacer is separating the tag from the metallic surface, no tests were performed in the course of this dissertation to ascertain the validity of this claim.

Another limitation of the current system is the number of RFID readers connected in the same bus. While four connected to the same processing unit is an improvement from the Balogh control interface (which connects with a maximum of two RWMs) currently used on Renault, it still pales in comparison with the ContriNET bus, capable of connecting 32 different readers. This leads to another major improvement necessary to make this solution more viable: reduce the time necessary to receive a message in the ESP32 UART. The current implementation reads the messages from RFID readers byte by byte, with each byte read on a different cycle of the application loop. This essentially causes one byte to be read every 1 millisecond, which in turn forces messages to be read only every 70 milliseconds. Initially messages were processed in the same loop, but this method proved unreliable. Discovering the cause of this problem would allow messages to be read at the 115200 baudrate the UART is configured with.

Major improvements can also be achieved in the graphical interface. More information about the status of the shop floor could be displayed, namely about the machining and leak tester stations, further digitizing several of the line's assets. Another envisioned feature was the substitution of the GALIA paper tags with RFID tags. While Renault has preference for the GALIA tag, as it is the standard for Groupe Renault, and also very cheap, it presents the same problems as normal paper tags. Replacing them with an RFID tag would allow to store all necessary information, and have it be more readily available when the containers reach their destination.

Bibliography

[1] Zhong RY, Xu X, Klotz E, Newman ST. Intelligent Manufacturing in the Context of Industry 4.0: A Review. Engineering. 2017 oct;3(5):616–630. Available from: https://www.sciencedirect.com/science/article/pii/S2095809917307130.

[2] Pinho FAS. Automatização da Traçabilidade das Linhas de Carters na Renault Cacia. Universidade de Coimbra. Coimbra; 2012. Available from: https://estudogeral.sib.uc.pt/jspui/handle/10316/20556.

[3] Groupe Renault. Renault sites across the world; [cited 2018-03-25]. Available from: https://group.renault.com/en/our-company/locations/our-industrial-locations/.

[4] Renault. Renault Cacia. Renault; [cited 2018-02-27]. Available from: https://www.renault.pt/descubra-a-renault/cacia/.

[5] Aluminiumleader. Aluminium applications - Transport; [cited 2018-03-05]. Available from: https://www.aluminiumleader.com/application/transport/.

[6] Isaac Maw. Automotive Aluminum Demand Growing: What It Means for Tier Ones; [cited 2018-03-05]. Available from: https://www.engineering.com/AdvancedManufacturing/ArticleID/16394/Automotive-Aluminum-Demand-Growing-What-It-Means-for-Tier-Ones.aspx.

[7] MyZebra. ZEBRA 110PAX4; 2018 [cited 2018-08-26]. Available from: http://www.myzebra.com.pt/pt/zebra-110pax4/1850-zebra-112el3e-00000.html.

[8] Tataru O. Melhoria da rastreabilidade dos componentes da caixa de velocidades. Universidade de Aveiro. Aveiro; 2013. Available from: https://ria.ua.pt/bitstream/10773/11549/1/7987.pdf.

[9] Balogh SA. OMX Tag Series;. Available from: https://www.baloghusa.com/{_}datasheets/ds{_}tags/OMXDATA.pdf.

[10] Balogh SA. BIDP-170/** ProfiBus-DP ® Control Interface;. Available from: https://www.baloghusa.com/{_}datasheets/ds{_}control{_}interfaces/BIDP-170DATA.pdf.

[11] Balogh SA. BIDP 170 - Profibus Specifications;. Available from: https://www.baloghusa.com/{_}manuals/man{_}control{_}interfaces/bidp{_}170{_}man.pdf.

[12] Siemens AG. PROFIBUS Network Manual; 2009. Available from: http://stest1.etnetera.cz/ad/current/content/data{_}files/automatizacni{_}systemy/prumyslova{_}komunikace/profibus/sysman{_}profibus-network-manual{_}2009-04{_}en.pdf.

[13] Lambert DM, Cooper MC. Issues in Supply Chain Management. Industrial Marketing Management. 2000 jan;29(1):65–83. Available from: https://www.sciencedirect.com/science/article/pii/S0019850199001133?via{%}3Dihub.

[14] Hugos MH. Essentials of Supply Chain Management. 4th ed. New Jersey; 2018.

[15] GS1. The GS1 Traceability Standard - What you need to know; 2006. Available from: https://www.gs1.org/docs/traceability/GS1{_}tracebility{_}what{_}you{_}need{_}to{_}know.pdf.

[16] Stasa P, Benes F, Svub J, Kang YS, Unucka J, Vojtech L, et al. Ensuring the Visibility and Traceability of Items through Logistics Chain of Automotive Industry based on AutoEPCNet Usage. 2016;Available from: http://advances.utc.sk/index.php/AEEE/article/viewFile/1788/1169.

[17] Santos C, Mehrsai A, Barros AC, Araújo M, Ares E. Towards Industry 4.0: an overview of European strategic roadmaps. Procedia Manufacturing. 2017 jan;13:972–979. Available from: https://www.sciencedirect.com/science/article/pii/S235197891730728X.

[18] Lu Y. Industry 4.0: A survey on technologies, applications and open research issues. Journal of Industrial Information Integration. 2017 jun;6:1–10. Available from: https://www.sciencedirect.com/science/article/pii/S2452414X17300043{#}bib0073.

[19] Roblek V, Meško M, Krapež A. A Complex View of Industry 4.0. SAGE Open. 2016;Available from: http://journals.sagepub.com/doi/pdf/10.1177/2158244016653987.

[20] GS1. GS1 General Specifications GS1-128 Symbology Specifications; [cited 2018-03-05]. Available from: http://www.gs1tw.org/twct/gs1w/download/GS{_}Section{_}5-3{_}V7.pdf.

[21] QRcode. Information capacity and versions of QR Code. QRcode; [cited 2018-03-01]. Available from: http://www.qrcode.com/en/about/version.html.

[22] Barcodes Inc . Honeywell; [cited 2018-03-07]. Available from: https://www.barcodesinc.com/honeywell/part-1450g1d-2usb-1.htm{#}specs.

[23] Keyence. SR-1000; [cited 2018-08-22]. Available from: https://www.keyence.com/products/vision/barcode/sr-1000/models/sr-1000/index.jsp.

[24] GS. Global User Manual User guide to the main GS1 identification and barcoding standards Log of Changes; [cited 2018-03-03]. Available from: https://www.gs1.org.sg/Portals/0/GS1/DownloadFiles/Global{_}User{_}Manual2017.pdf.

[25] Keyence. What is a DataMatrix code?; [cited 2018-04-05]. Available from: https://www.keyence.com/ss/products/auto{_}id/barcode{_}lecture/basic{_}2d/datamatrix/.

[26] GS1. GS1 DataMatrix Guideline Overview and technical introduction to the use of GS1 DataMatrix; 2018. Available from: https://www.gs1.org/docs/barcodes/GS1{_}DataMatrix{_}Guideline.pdf.

[27] Keyence. What is a QR code?; [cited 2018-04-05]. Available from: https://www.keyence.com/ss/products/auto{_}id/barcode{_}lecture/basic{_}2d/qr/.

[28] QRcode. QR Code Model 1 and Model 2; [cited 2018-04-05]. Available from: http://www.qrcode.com/en/codes/model12.html.

[29] Datalogic. AREX 50; [cited 2018-10-16]. Available from: http://www.scanning2.datalogic.com/e-catalog/ec-AREX-50.aspx.

[30] Sic Marking. Dot Peen or Laser DataMatrix Marking; [cited 2018-04-05]. Available from: https://www.sic-marking.com/datamatrix-marking.

[31] Ostling Marking Systems. DataMatrix; [cited 2018-04-05]. Available from: https://www.ostling-markingsystems.com/products/datamatrix-code/sample-markings-datamatrix-code/.

[32] Pryor. Dot peen marking machine; [cited 2018-04-05]. Available from: http://www.directindustry.com/prod/pryor-marking-technology/product-33451-1814655.html.

[33] Zhu X, Mukhopadhyay SK, Kurata H. A review of RFID technology and its managerial applications in different industries. Journal of Engineering and Technology Management. 2012 jan;29(1):152–167. Available from: https://www.sciencedirect.com/science/article/pii/S092347481100049X.

[34] Klaus Finkenzeller. RFID Handbook. 3rd ed. John Wiley & Sons, Ltd.; 2010.

[35] El Khaddar MA, Boulmalf M, Harroud H, Elkoutbi M. RFID Middleware Design and Architecture. In: Designing and Deploying RFID Applications. InTech; 2011. Available from: http://www.intechopen.com/books/designing-and-deploying-rfid-applications/rfid-middleware-design-and-architecture.

[36] Lahiri S. RFID sourcebook. 3rd ed. New Jersey: IBM Press; 2006. Available from: http://opac.ua.pt/cgi-bin/koha/opac-detail.pl?biblionumber=221045.

[37] Hashemi A, Sarhaddi AH, Emami H. A Review on Chipless RFID Tag Design. Majlesi Journal of Electrical Engineering. 2013;7(2):68–75. Available from: http://search.ebscohost.com/login.aspx?direct=true{&}db=a9h{&}AN=94142757{&}lang=pt-br{&}site=ehost-live.

[38] Duroc Y, Tedjini S. RFID: A key technology for Humanity. Comptes Rendus Physique. 2018 jan;19(1-2):64–71. Available from: https://www.sciencedirect.com/science/article/pii/S1631070518300124.

[39] Omicron Lab. RFID Resonance Frequency and Quality Factor Measurement; [cited 2018-03-08]. Available from: https://www.omicron-lab.com/bode-100/application-notes-know-how/application-notes/rfid-resonance-frequency-measurement.html.

[40] RFID4U. RFID Regulations | RFID4U; [cited 2018-03-09]. Available from: http://rfid4u.com/rfid-basics-resources/basics-rfid-regulations/.

[41] GS1 Portugal - CODIPOR. GS1 EPCglobal;Available from: http://www.gs1pt.org/wp-content/uploads/2016/04/Standard{_}GS1{_}EPC.pdf.

[42] Kang YS, Kim H, Lee YH, Kang YS, Kim H, Lee YH. Implementation of an RFID-Based Sequencing-Error-Proofing System for Automotive Manufacturing Logistics. Applied Sciences. 2018 jan;8(1):109. Available from: http://www.mdpi.com/2076-3417/8/1/109.

[43] Bártolo RGL. Integração de sistemas de rastreabilidade em ambiente industrial. Aveiro; 2013. Available from: https://ria.ua.pt/handle/10773/12483.

[44] Almeida JCSCB. Novo sistema de rastreabilidade industrial. Universidade de Aveiro. Aveiro; 2012. Available from: http://ria.ua.pt/handle/10773/10110.

[45] Liu C, Jiang P. A Cyber-physical System Architecture in Shop Floor for Intelligent Manufacturing. Procedia CIRP. 2016 jan;56:372–377. Available from: https://www.sciencedirect.com/science/article/pii/S2212827116310514.

[46] Zhang Y, Zhang G, Wang J, Sun S, Si S, Yang T. Real-time information capturing and integration framework of the internet of manufacturing things. International Journal of Computer Integrated Manufacturing. 2015 aug;28(8):811–822. Available from: http://www.tandfonline.com/doi/full/10.1080/0951192X.2014.900874.

[47] Qu T, Yang HD, Huang GQ, Zhang YF, Luo H, Qin W. A case of implementing RFID-based real-time shop-floor material management for household electrical appliance manufacturers. Journal of Intelligent Manufacturing. 2012 dec;23(6):2343–2356. Available from: http://link.springer.com/10.1007/s10845-010-0476-2.

[48] Baluff Inc . Traceability in Manufacturing. Baluff Inc.; [cited 2018-03-02]. Available from: http://www.rwearl.com/Portals/9/pdf/Traceability.pdf.

[49] GlobeRanger Corporation. Mainpage; [cited 2018-03-13]. Available from: https://www.globeranger.com/.

[50] GlobeRanger Corporation. Manufacturing; [cited 2018-03-13]. Available from: https://www.globeranger.com/solutions/manufacturing/.

[51] Contrinex. RFID | Contrinex; [cited 2018-05-16]. Available from: https://www.contrinex.com/business-unit/rfid/.

[52] Chen PPS. The Entity-Relationship Model-Toward a Unified View of Data. Boston: Massachussetts Institute of Technology;. Available from: http://citeseerx.ist.psu.edu/viewdoc/download;jsessionid=48D27EB875931869637FD9C336914B87?doi=10.1.1.123.1085{&}rep=rep1{&}type=pdf.

[53] Contrinex. RLS-1183-020 Datasheet; 2018. Available from: https://www.contrinex.com/product/rls-1183-020/.

[54] Espressif Systems. ESP32 Series Datasheet Including; 2018. Available from: www.espressif.com/en/subscribe.

Appendices

Appendix A

Graphical Interface

Besides the elements described and shown in section 5.7, the graphical interface developed in web pages has several other components. Namely, a graph that shows the production over time, as shown in the following figure:

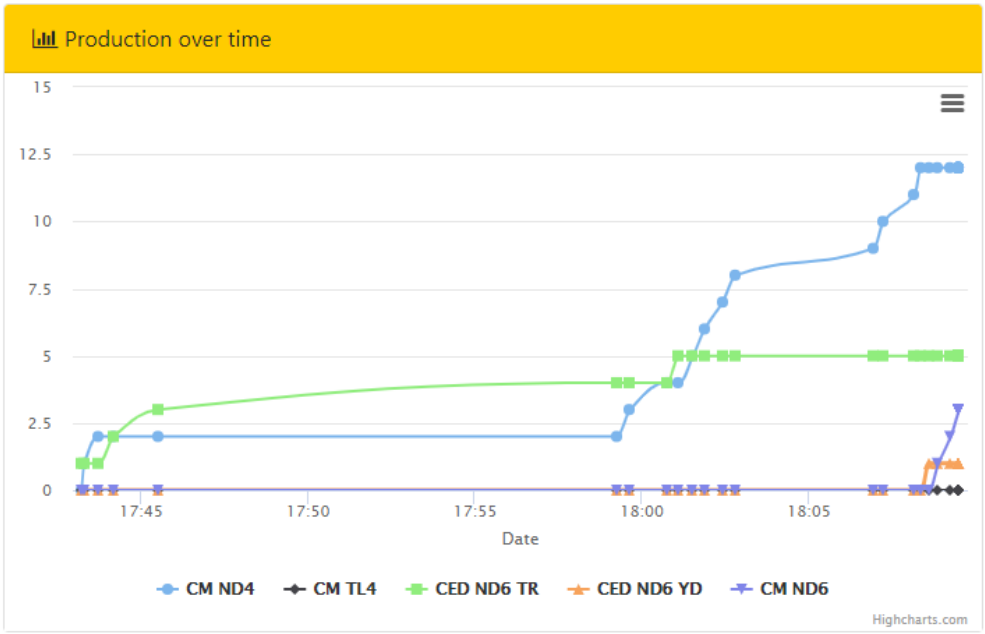


Figure A.1: Graphical interface - Production over time chart

This graph shows the detailed information about the different references produced, alongside the date (hour and day) the housing was manufactured. The graph is automatically updated whenever a new housing is produced. To access this information, pages containing this graph perform HTTP requests with the AJAX method. The page it sends these requests to is called RFID_line.php, which responds with data in JSON format.

As the figure shows,the same request is being used to update two different charts: one shows the production over time, another simply shows the total production of each reference. This second graph can be visible in figure A.2.

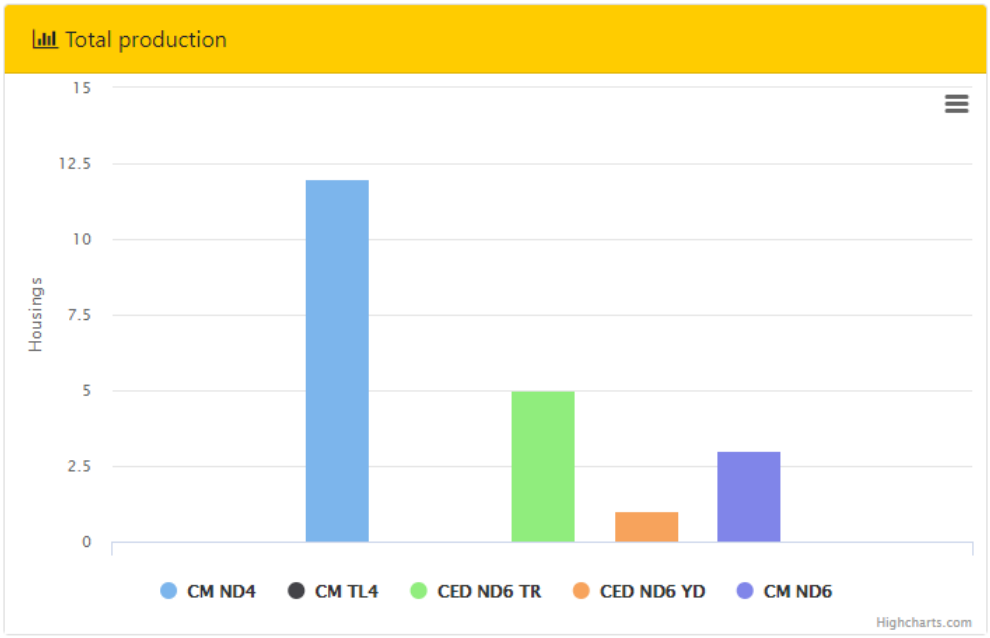


Figure A.2: Graphical interface - Total production chart

The two charts are updated through two different methods. Each of the series of the total production chart is defined by one single number: the total of housings produced of a given reference. Therefore, if this number increases, the series is replaced entirely, and the graph is updated. However, the production over time chart has its series defined by multiple values of time and total housings. Replacing the entire series by the most recent answer from the server is possible, however this unnecessarily adds processing time to the highcharts tool, which has to completely redraw a high number of data. Therefore the addPoint method was used. The last value of the server's response is the point added to the graph. This addPoint function only adds the point if it is different from the points already present in the series. Therefore only a real increase in the number of housings produced causes a point to be added to the chart.

These charts are used, in conjunction with the other elements mentioned in section 5.7, in several different pages. The intent was to give the user access to relevant information without having to change the page, having to merely scroll down.

The following figure shows the entirety of the Raw Products Nonconformities page. In a computer screen only a part of this interface is visible in a browser tab. The user scrolls down the page to see the other elements of the page.

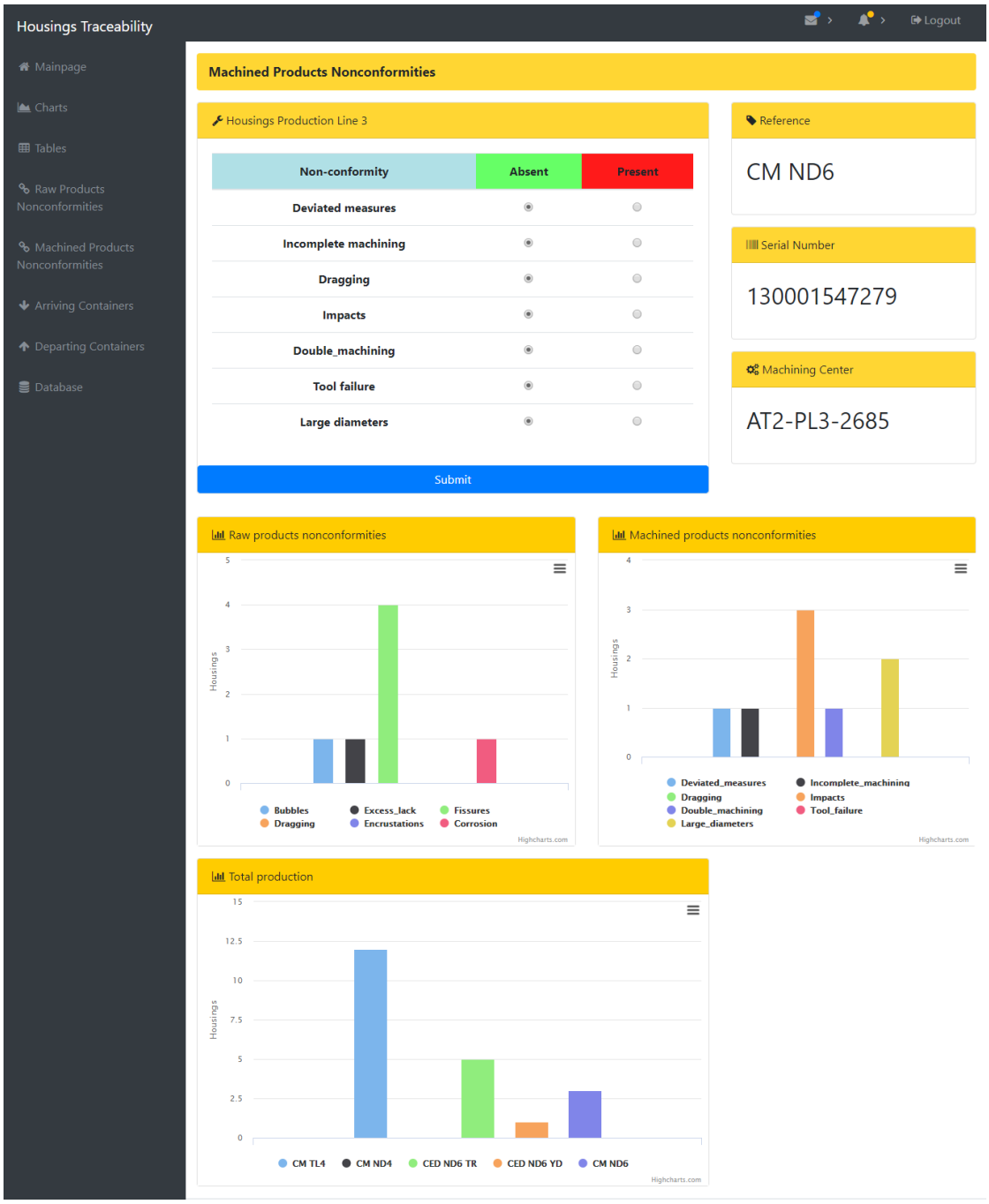


Figure A.3: Graphical interface - phpMyAdmin interface

As can be seen, pages are built vertically, allowing the user to access several elements of the interface in a single page, without, however, overburdening the page with information, separating each page with a distinct function in the interface.

The remaining pages which contain multiple elements throughout several rows are:

Mainpage:

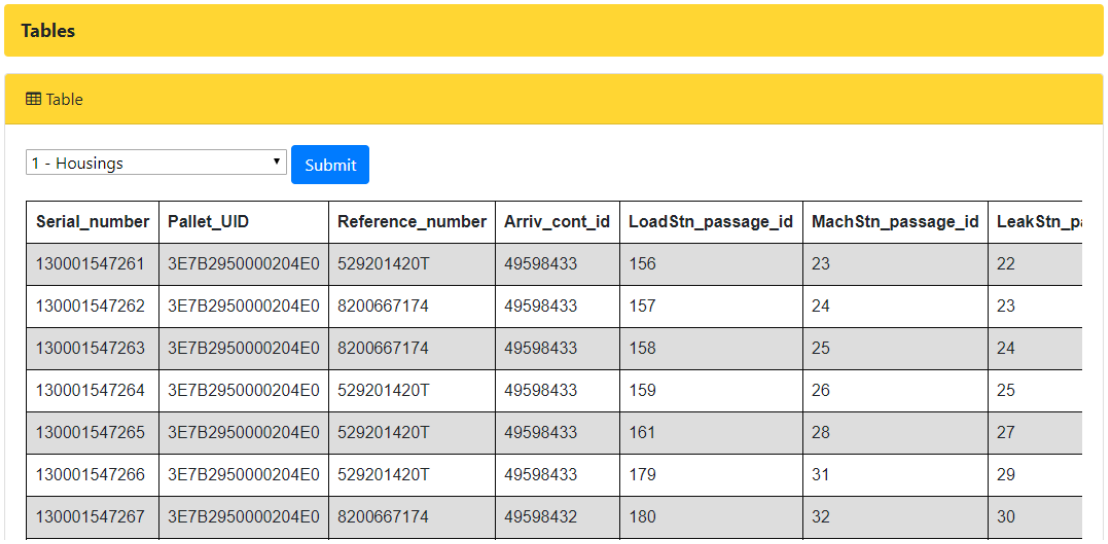
- Production line schematic, alongside buttons for raw and machined nonconformities

stations, as shown in figure 5.17.
- Production over time chart, as shown in figure A.1.
- Total production chart, shown in previous figure A.2.
- Nonconformities charts, side by side, as shown in figure A.3

Charts:
-Contains all the charts in the mainpage, without having a line schematic and buttons for nonconformities pages.

Machined Products Nonconformities:
- Visually it is very similar to the Raw Products Nonconformities page. Its visual test table is different, with a list of the eight possible machined nonconformities. It too has two cards showing the reference and serial number, though obviously they show the information regarding the housing in the last station.

The two remaining pages are called Tables and Database. These have very different interfaces from each other and the previous pages. The Tables page is illustrated bellow.



Serial_number	Pallet_UID	Reference_number	Arriv_cont_id	LoadStn_passage_id	MachStn_passage_id	LeakStn_p
130001547261	3E7B2950000204E0	529201420T	49598433	156	23	22
130001547262	3E7B2950000204E0	8200667174	49598433	157	24	23
130001547263	3E7B2950000204E0	8200667174	49598433	158	25	24
130001547264	3E7B2950000204E0	529201420T	49598433	159	26	25
130001547265	3E7B2950000204E0	529201420T	49598433	161	28	27
130001547266	3E7B2950000204E0	529201420T	49598433	179	31	29
130001547267	3E7B2950000204E0	8200667174	49598432	180	32	30

Figure A.4: Graphical interface - Tables

This page exists so the user can look through the several tables, allowing him to select the table they want loaded onto the page. This page has its limitations, particularly the lack of a search tool. Several attempts were made at integrating open source search code for tables. However, results were never satisfactory.

Instead the Database page is used for that functionality, in conjunction with the ability of creating tables or altering them. This page uses the phpMyAdmin sofware, specifically designed to interact with MySQL databases. This page can be protected by a user name and password, in case the access to the database is intended to be limited to few people. The login can be seen on figure A.5.

Figure A.5: Graphical interface - phpMyAdmin login

Once the login is successful the user can access the database. While possible, the user shouldn't need to manually enter or alter any value in this database, as it is automatically updated by the data coming from the ESP32. However, the user can create new tables and columns, as well as search for specif values in all tables. Whenever a foreign keys is present in a table, values of those columns are linked to the tables where the keys are primary keys. For example, the user can click on a Arriving_container_ID on the Housings table and be immediately brought to the Arriving Container Table, where the remaining attributes of that container can be viewed.

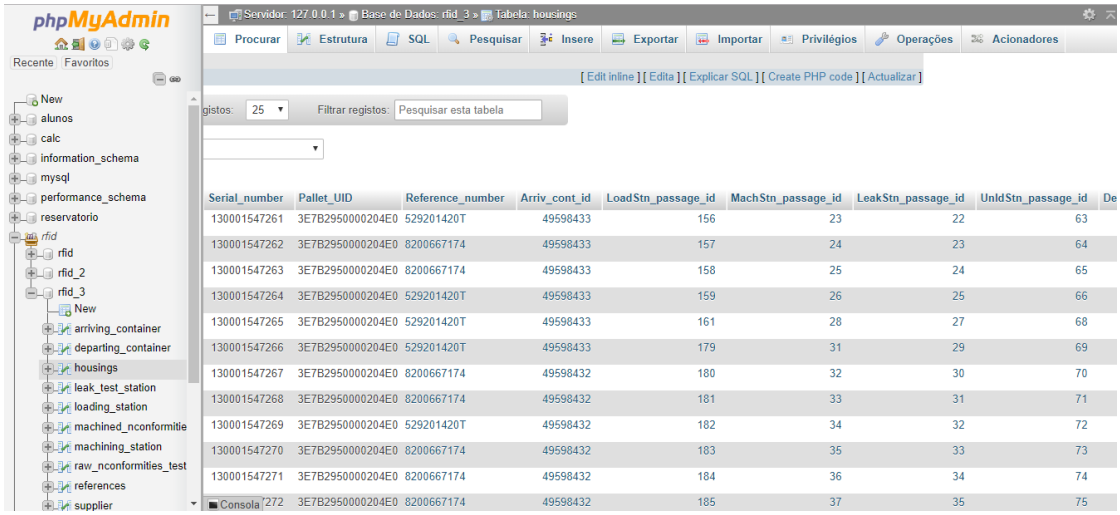


Figure A.6: Graphical interface - phpMyAdmin interface

The interface was also built to adapt to smartphone screens. Bootstrap allows the designer of the page to specify how each element of the page adapts to screens of different sizes. Smartphones are defined as small screens in Bootstrap, so any element meant to have a specific width in a smartphone screen must be defined using the sm prefix. The

code for this can be seen on the following figure.

```
514         <div class="col-sm-4">
515
516                 <div class="card mb-3">
517               <div class="card-header" style="background-color:#ffd633;">
518                 <i class="fa fa-tag"></i> Reference</div>
519               <div class="card-body">
520                 <p><font size="6"><span id="Reference_html"> <?PHP echo $Reference;?></span></font></p>
521
522               </div>
523             </div>
524
525             <div class="card mb-3">
526               <div class="card-header" style="background-color:#ffd633;">
527                 <i class="fa fa-barcode"></i> Serial Number</div>
528               <div class="card-body">
529 <p><font size="6"><span id="Serial_number_html"> <?PHP echo $Serial_number;?></span></font></p>
530
531               </div>
532             </div>
533
534
535
536         </div>
```

Figure A.7: Graphical interface - Code used to create interface divisions

This code creates the Reference and Serial Number cards visible in figure A.8. In the absence of the line 514 present in the code, each division of the would span the entire width of the device, with each element stacked vertically.

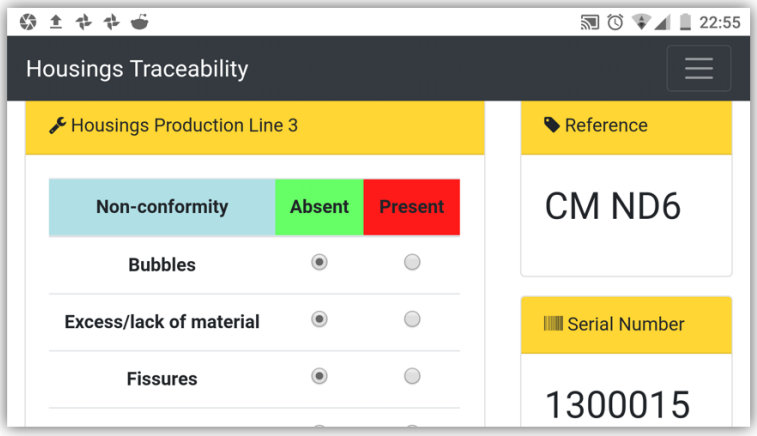


Figure A.8: Graphical interface - Horizontal smartphone interface (partial)

One major difference between the vertical and horizontal display is that, given the small width of a smartphone while held vertically, interface elements are indeed showed on top one another. The result can be seen on figure A.9.

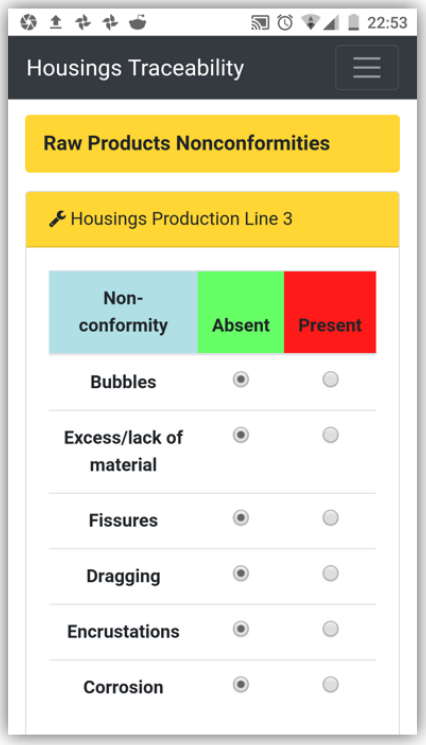


Figure A.9: Graphical interface - Vertical smartphone interface (partial)

For the cards in the interface to retain their proportions in the vertical display each division would need to be defined as 'col-dimension' instead of 'col-sm-dimension'. However, as can be expected, elements such as the Reference and Serial Number cards would be stretched vertically in order to fit in a third of the screen's width.

Figure A.10 shows the entire vertical display of the interface (separated in two parts simply, with the right being the continuation of the page shown on the left).

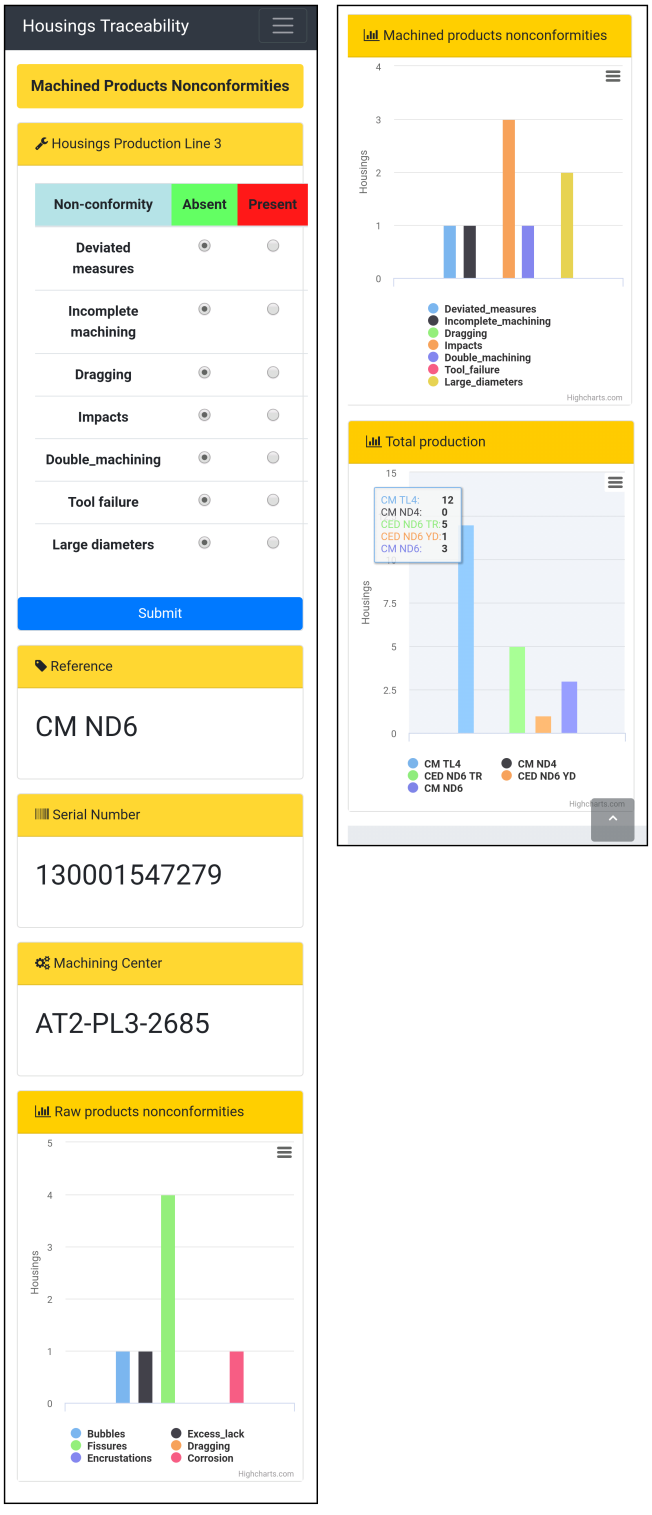


Figure A.10: Graphical interface - Vertical smartphone interface (complete)

Appendix B

GALIA tags

Figure B.1: GALIA from external supplier

Figure B.2: GALIA from internal supplier (other Renault facilities)

Appendix C

Developed PCB

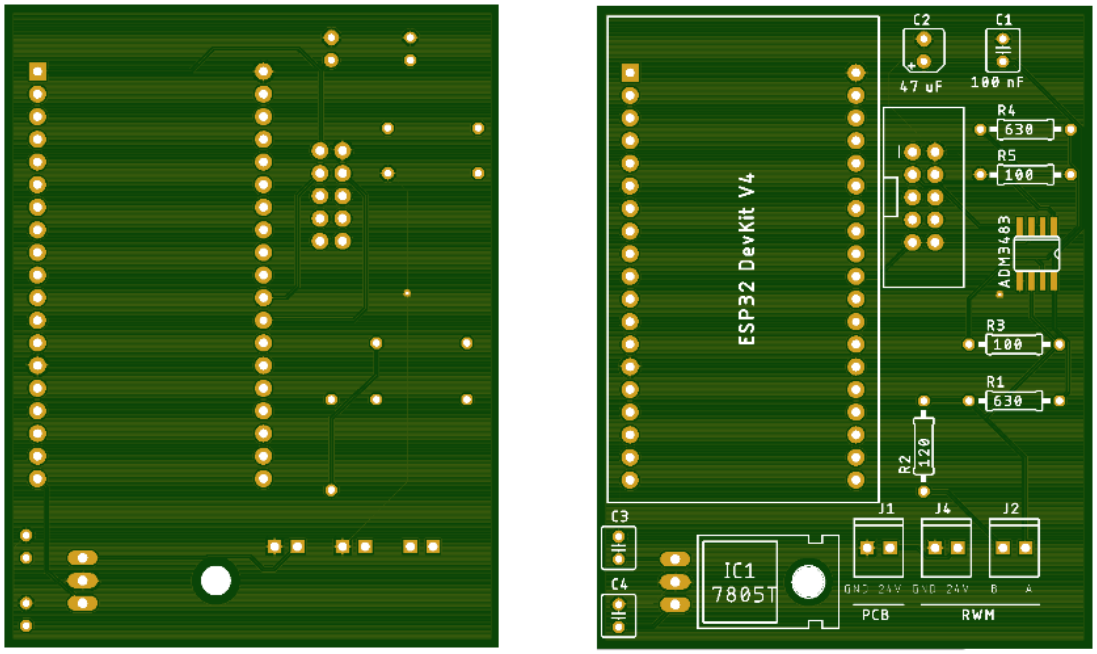


Figure C.1: Bottom and top layers of developed PCB